



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 № 16**

Дисциплина Функциональное и логическое программирование

Тема Использование правил в программе на Prolog

Студент Ильясов И. М.

Группа ИУ7-63Б

Оценка (баллы) \_\_\_\_\_

Преподаватель Толпинская Н. Б., Строганов Ю. В.

Москва, 2020 г.

**Цель работы** – изучить использование правил в программе: структуру, особенности оформления, а также, способ и принципы выполнения таких программ на Prolog.

**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**

**Создать базу знаний: «ПРЕДКИ»**, позволяющую **наиболее эффективным** способом (за меньшее количество шагов, что обеспечивается меньшим количеством предложений БЗ - правил), используя разные варианты (примеры) **одного вопроса**, определить (указать: какой вопрос для какого варианта):

1. по имени субъекта определить всех его бабушек (предки 2-го колена),
2. по имени субъекта определить всех его дедушек (предки 2-го колена),
3. по имени субъекта определить всех его бабушек и дедушек (предки 2-го колена),
4. по имени субъекта определить его бабушку по материнской линии (предки 2-го колена),
5. по имени субъекта определить его бабушку и дедушку по материнской линии (предки 2-го колена).

Минимизировать количество правил и количество вариантов вопросов.

Использовать **конъюнктивные правила и простой вопрос**.

**Для одного из вариантов ВОПРОСА** и конкретной БЗ **составить таблицу**, отражающую конкретный порядок работы системы, с объяснениями:

очередная проблема на каждом шаге и метод ее решения;

каково новое текущее состояние резольвенты, как получено;

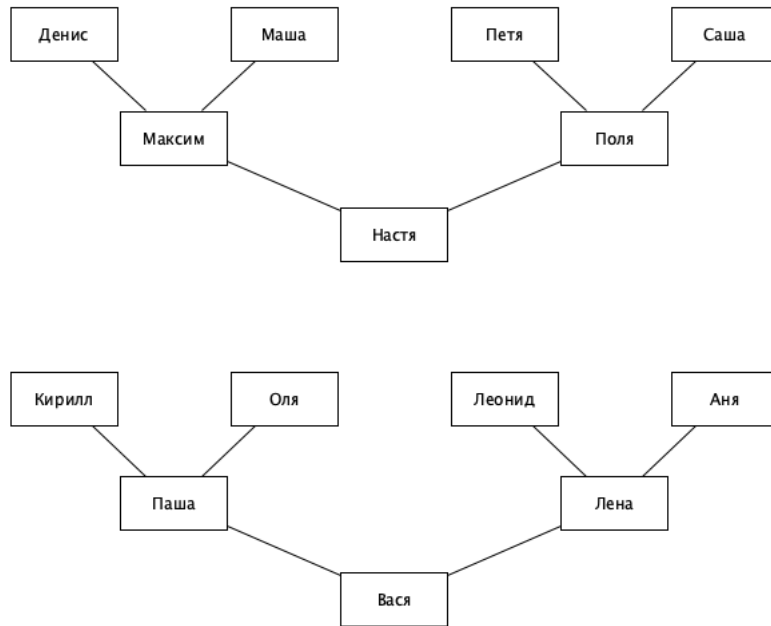
какие дальнейшие действия? (Запускается ли алгоритм унификации? Каких термов?

Почему этих?);

вывод по результатам очередного шага и дальнейшие действия.

Т.к. резольвента хранится в виде стека, то состояние резольвенты требуется отображать в столбик: вершина – сверху! Новый шаг надо начинать с нового состояния резольвенты!

## Генеалогическое представление семьи:



## Текст программы

### domains

father, mother, child = symbol.

### predicates

create\_family(father, mother, symbol)

get\_grandpars(symbol P, symbol GP1, symbol GM1, symbol GP2, symbol GM2)

### clauses

create\_family(denis, masha, max).

create\_family(petya, sasha, polya).

create\_family(max, polya, nastya).

create\_family(kirill, olya, pasha).

create\_family(leonid, anya, lena).

create\_family(pasha, lena, vasya).

get\_grandpars(P, GP1, GM1, GP2, GM2) :-

create\_family(F, M, P),

create\_family(GP1, GM1, F),

create\_family(GP2, GM2, M).

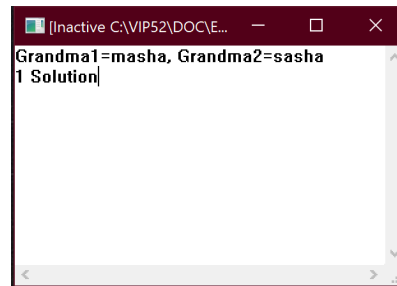
### goal

get\_grandpars(nastya, \_, Grandma1, \_, Grandma2).

## Примеры работы программы

1. По имени субъекта определить всех его бабушек (предки 2-го колена).

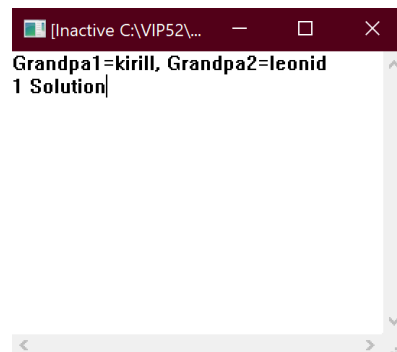
Ниже приведен ответ на вопрос о всех бабушках Насти.



```
[Inactive C:\VIP52\DOC\E...  —  □  ×  
Grandma1=masha, Grandma2=sasha  
1 Solution|
```

2. По имени субъекта определить всех его дедушек (предки 2-го колена).

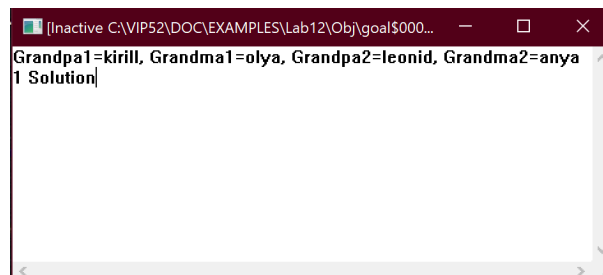
Ниже приведен ответ на вопрос о всех дедушках Васи.



```
[Inactive C:\VIP52\...  —  □  ×  
Grandpa1=kirill, Grandpa2=leonid  
1 Solution|
```

3. По имени субъекта определить всех его бабушек и дедушек (предки 2-го колена).

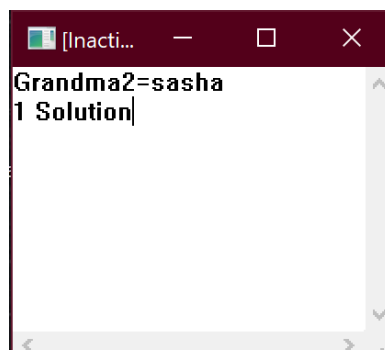
Ниже приведен ответ на вопрос о всех бабушках и дедушках Васи.



```
[Inactive C:\VIP52\DOC\EXAMPLES\Lab12\Obj\goal$000...  —  □  ×  
Grandpa1=kirill, Grandma1=olya, Grandpa2=leonid, Grandma2=anya  
1 Solution|
```

4. По имени субъекта определить его бабушку по материнской линии (предки 2-го колена).

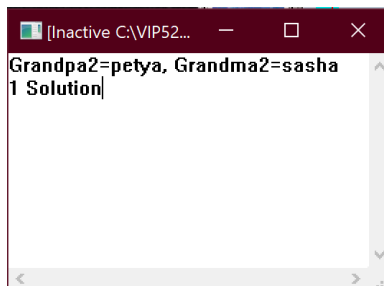
Ниже приведен ответ на вопрос о бабушке Насти по материнской линии.



```
[Inacti...  —  □  ×  
Grandma2=sasha  
1 Solution|
```

5. По имени субъекта определить его бабушку и дедушку по материнской линии (предки 2-го колена).

Ниже приведен ответ на вопрос о бабушке и дедушке Насти по материнской линии.



```
[Inactive C:\VIP52...]
```

```
Grandpa2=petya, Grandma2=sasha
```

```
1 Solution
```

### Задание с таблицей

`get_grandpars(nastya, _, Grandma1, _, Grandma2)` – по имени субъекта имена всех бабушек.

№ шага	Состояние резольвенты, и вывод: дальнейшие действия (почему?)	Для каких термов запускается алгоритм унификации: T1=T2 и каков <b>результат</b> (и подстановка)	Дальнейшие действия: прямой ход или откат (почему и к чему приводит?)
1	<code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code>	T1 = <code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code> . T2 = <code>create_family(denis, masha, max)</code> . Неудача, разные функторы.	Прямой ход к следующему предложению.
2	<code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code>	T1 = <code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code> . T2 = <code>create_family(petya, sasha, polya)</code> . Неудача, разные функторы.	Прямой ход к следующему предложению.
3	<code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code>	T1 = <code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code> . T2 = <code>create_family(max, polya, nastya)</code> . Неудача, разные функторы.	Прямой ход к следующему предложению.
4	<code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code>	T1 = <code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code> . T2 = <code>create_family(kirill, olya, pasha)</code> . Неудача, разные функторы.	Прямой ход к следующему предложению.
5	<code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code>	T1 = <code>get_grandpars(nastya, _, Grandma1, _, Grandma2)</code> . T2 = <code>create_family(leonid, anya, lena)</code> . Неудача, разные функторы.	Прямой ход к следующему предложению.

6	get_grandpars(nastya, _, Grandma1, _, Grandma2)	T1 = get_grandpars(nastya, _, Grandma1, _, Grandma2). T2 = create_family(pasha, lena, vasya). Неудача, разные функторы.	Прямой ход к следующему предложению.
7	get_grandpars(nastya, _, Grandma1, _, Grandma2)	T1 = get_grandpars(nastya, _, Grandma1, _, Grandma2) T2 = get_grandpars(P, GP1, GM1, GP2, GM2). Успех, подстановка {nastya=X, Grandma1=GM1, Grandma2=GM2}.	Прямой ход к сопоставлению family(F, M, nastya), поиск с начала предложений.
8	create_family(F, M, nastya), create_family(_, GM1, F), create_family(_, GM2, M)	T1 = create_family(F, M, nastya), T2 = create_family(denis, masha, max). Неудача, nastya != max.	Прямой ход к следующему предложению.
9	create_family(F, M, nastya), create_family(_, GM1, F), create_family(_, GM2, M)	T1 = create_family(F, M, nastya), T2 = create_family(petya, sasha, polya). Неудача, nastya != polya.	Прямой ход к следующему предложению.
10	create_family(F, M, nastya), create_family(_, GM1, F), create_family(_, GM2, M)	T1 = create_family(F, M, nastya), T2 = create_family(max, polya, nastya). Успех, подстановка {F=max, M=polya, nastya=nastya}.	Прямой ход к сопоставлению create_family(_, GM1, max), поиск с начала предложений.
11	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = create_family(denis, masha, max). Успех, подстановка {GM1=masha, max=max}.	Прямой ход к сопоставлению create_family(_, GM2, polya), поиск с начала предложений.
12	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = create_family(denis, masha,	Прямой ход к следующему

		max). Неудача, polya != max.	предложению.
13	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = create_family(petya, sasha, polya). Успех, подстановка {GM2=sasha, polya=polya}.	<b><u>Вывод: GM1=masha, GM2=sasha.</u></b> Прямой ход к следующему предложению, реконкретизация GM2.
14	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = create_family(max, polya, nastya). Неудача, polya != nastya.	Прямой ход к следующему предложению.
15	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = create_family(kirill, olya, pasha). Неудача, polya != pasha.	Прямой ход к следующему предложению.
16	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = create_family(leonid, anya, lena). Неудача, polya != lena.	Прямой ход к следующему предложению.
17	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = create_family(pasha, lena, vasya). Неудача, polya != vasya.	Прямой ход к следующему предложению.
18	create_family(_, GM2, polya)	T1 = create_family(_, GM2, polya), T2 = get_grandpars(P, GP1, GM1, GP2, GM2). Неудача, разные функторы.	Откат, переход к предыдущему состоянию резольвенты (шаг 11), реконкретизация GM1.
19	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = create_family(petya, sasha, polya). Неудача, max != polya.	Прямой ход к следующему предложению.

20	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = create_family(max, polya, nastya). Неудача, max != nastya.	Прямой ход к следующему предложению.
21	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = create_family(kirill, olya, pasha). Неудача, max != pasha.	Прямой ход к следующему предложению.
22	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = create_family(leonid, anya, lena). Неудача, max != lena.	Прямой ход к следующему предложению.
23	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = create_family(pasha, lena, vasya). Неудача, max != vasya.	Прямой ход к следующему предложению.
24	create_family(_, GM1, max), create_family(_, GM2, polya)	T1 = create_family(_, GM1, max), T2 = get_grandpars(P, GP1, GM1, GP2, GM2). Неудача, разные функторы.	Откат, переход к предыдущему состоянию резольвенты (шаг 10), реконкретизация F и M.
25	create_family(F, M, nastya), create_family(_, GM1, F), create_family(_, GM2, M)	T1 = create_family(F, M, nastya), T2 = create_family(kirill, olya, pasha). Неудача, nastya != pasha.	Прямой ход к следующему предложению.
26	create_family(F, M, nastya), create_family(_, GM1, F), create_family(_, GM2, M)	T1 = create_family(F, M, nastya), T2 = create_family(leonid, anya, lena). Неудача, nastya != lena.	Прямой ход к следующему предложению.
27	create_family(F, M, nastya), create_family(_, GM1, F),	T1 = create_family(F, M, nastya), T2 = create_family(pasha, lena, vasya). Неудача, nastya != vasya.	Прямой ход к следующему предложению.



	create_family(_, GM2, M)		
28	create_family(F, M, nastya), create_family(_, GM1, F), create_family(_, GM2, M)	T1 = create_family(F, M, nastya), T2 = get_grandpars(P, GP1, GM1, GP2, GM2) Неудача, разные функторы.	Откат, переход к предыдущему состоянию резольвенты (шаг 7).
29	get_grandpars(nastya, _, GM1, _, GM2); конец clauses; опустошение резольвенты; завершение работы.		

## **Ответы на вопросы**

### **1) В каком случае система запускает алгоритм унификации? (Как эту необходимость на формальном уровне распознает система?)**

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

### **2) Каковы назначение и результат использования алгоритма унификации?**

Назначение алгоритма унификации заключается в попарном сопоставлении термов и попытке построить для них общий пример. Унификация может завершаться успехом или тупиковой ситуацией (неудачей).

### **3) Какое первое состояние резольвенты?**

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

### **4) 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).

### **5) 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.

### **6) Как применяется подстановка, полученная с помощью алгоритма унификации?**

Применение подстановки  $\{X_1=T_1, \dots, X_n=T_n\}$  заключается в замене каждого вхождения переменной  $X_i$  на соответствующий терм  $T_i$ .

### **7) 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 знания остались не отмеченные предложения.