|  |  |
| --- | --- |
| 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 № 13**

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

Москва, 2020 г.

**Цель работы** – получить навыки построения модели предметной области, разработки и оформления программы на Prolog, изучить принципы, логику формирования программы и отдельные шаги выполнения программы на Prolog.

**Задачи работы**:

* приобрести навыки декларативного описания предметной области с использованием фактов и правил;
* изучить способы использования термов, переменных, фактов и правил в программе на Prolog, принципы и правила сопоставления и отождествления, порядок унификации.

**Задание лабораторной работы**

Составить программу, т.е. модель предметной области – базу знаний, объединив в ней информацию – знания:

* **«Телефонный справочник»:** Фамилия, №тел, Адрес – структура (Город, Улица, №дома, №кв),
* **«Автомобили»:** Фамилия\_владельца,Марка, Цвет, Стоимость, и др.,
* **«Вкладчики банков»:** Фамилия, Банк, счет, сумма, др.

Владелец может иметь несколько телефонов, автомобилей, вкладов (Факты).

Используя правила, обеспечить возможность поиска:

1. **а)** По № телефона найти: Фамилию, Марку автомобиля, Стоимость автомобиля (может быть несколько),

**в)** Используя сформированное в пункте **а)** правило, по № телефона найти: только Марку автомобиля (автомобилей может быть несколько),

1. Используя простой, не составной вопрос: по Фамилии (уникальна в городе, но в разных городах есть однофамильцы) и Городу проживания найти: Улицу проживания, Банки, в которых есть вклады и №телефона.

**Для задания1 и задания2**:

Для одного из вариантов ответов, и для **а)** и для **в)**, **описать словесно** порядок поиска ответа на вопрос, указав, как выбираются знания, и, при этом, **для каждого этапа унификации, выписать подстановку** – наибольший общий унификатор, **и соответствующие примеры** термов.

**Текст программы**

domains

surname, phone, city, street = string.

house, flat = integer.

address = address(city, street, house, flat).

car\_mark, car\_color = string.

car\_cost = integer.

bank\_name, bank\_cardnumber = string.

bank\_sum = integer.

predicates

abonement(surname, phone, address).

car(surname, car\_mark, car\_color, car\_cost, city).

deposit(surname, bank\_name, bank\_cardnumber, bank\_sum, city).

find(phone, car\_mark, car\_cost).

find(phone, car\_mark).

clauses

abonement("Gorbunov", "89251472838", address("Korolev", "Glavnaya", 55, 122)).

abonement("Ilyasov", "89969503880", address("Moscow", "Severnaya", 12, 75)).

abonement("Sidenko", "89691929395", address("Moscow", "Semenovskaya", 25, 53)).

abonement("Stepanov", "185818582839", address("Korolev", "Krilatskaya", 12, 155)).

abonement("Gorbunov", "165615253616", address("Saint-Petersburg", "Nevskaya", 51, 122)).

car("Gorbunov", "Audi", "White", 2500000, "Korolev").

car("Ilyasov", "Mitsubishi", "Red", 800000, "Moscow").

car("Ilyasov", "Tesla", "Black", 6000000, "Moscow").

car("Sidenko", "Tesla", "Black", 6000000, "Moscow").

car("Sidenko", "Mercedes", "White", 3000000, "Moscow").

car("Stepanov", "Mercedes", "Black", 5000000, "Korolev").

car("Stepanov", "Audi", "White", 2500000, "Korolev").

car("Stepanov", "Mercedes", "White", 3000000, "Korolev").

car("Gorbunov", "Toyota", "White", 600000, "Saint-Petersburg").

deposit("Gorbunov", "Sberbank", "123456789", 30000, "Korolev").

deposit("Ilyasov", "Sberbank", "987654321", 20000, "Moscow").

deposit("Sidenko", "Tinkoff", "135798642", 60000, "Moscow").

deposit("Stepanov", "Alfa", "156273727", 20000, "Korolev").

deposit("Gorbunov", "Sberbank", "563281726", 100000, "Saint-Petersburg").

find(Phone, Car\_Mark, Car\_Cost) :- abonement(Surname, Phone, address(City, \_, \_, \_)),

car(Surname, Car\_Mark, \_, Car\_Cost, City).

find(Phone, Car\_Mark) :- find(Phone, Car\_Mark, \_).

goal

% LR13

% Task 1 (a)

find("89691929395", Car\_Mark, Car\_Cost).

% Task 2 (в)

find("89691929395", Car\_Mark).

% Task 3

City = "Korolev",

Surname = "Gorbunov",

abonement(Surname, Phone, address(City, Street, \_, \_)),

deposit(Surname, Bank\_Name, \_, \_, City).

**Примеры работы программы**

На рисунке 1 приведен ответ на вопрос о марке машины и цены машины по номеру телефона.

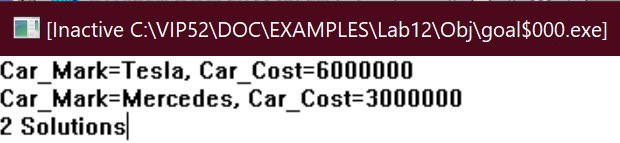


Рисунок 1. Ответ на вопрос о марке и цене машины по номеру владельца.

На рисунке 2 приведен ответ на вопрос только о марке машины по номеру телефона.



Рисунок 2. Ответ на вопрос только о марке машины по номеру владельца.

На рисунке 3 приведен ответ на простой вопрос об улице проживания, банках, в которых есть вклады, и номере телефона по фамилии и городу человека.

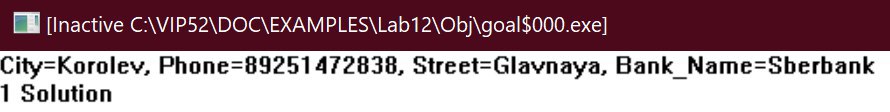


Рисунок 3. Ответ на вопрос об информации по человеку по фамилии и городу.

1. find("89251472838", Car\_Mark, Car\_Cost) – ненужные промежуточные шаги опущены.

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; результат; подстановка, если есть | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
|  | … | … |
| 1 | Производится сравнение find(“89251472838”, Car\_Mark, Car\_Cost). и find(Phone, Car\_Mark, Car\_Cost). Phone конкретизируется с “89251472838”. Связываются Car\_Mark с Car\_Mark и Car\_Cost с Car\_Cost.  Результат – find(“89251472838”, Car\_Mark, Car\_Cost) | Прямой ход, производится переход к унификации abonement(Surname, Phone, address(City, \_, \_, \_)), где переменная Phone равна “89251472838”. |
|  | … | … |
| 2 | Производится сравнение abonement(Surname, “89251472838”, address(City, \_, \_, \_)). и ("Gorbunov", "89251472838", address("Korolev", "Glavnaya", 55, 122)). Surname конкретизируется с “Gorbunov”, City – с “Korolev”, сравниваются номера.  Результат – abonement("Gorbunov", "89251472838", address("Korolev", \_, \_, \_)). | Прямой ход, производится переход к унификации car(Surname, Car\_Mark, \_, Car\_Cost, City). При этом Phone = “89251472838”, Surname = “Gorbunov”, City = “Korolev”. |
|  | … | … |
| 3 | Производится сравнение car(“Gorbunov”, Car\_Mark, \_, Car\_Cost, “Korolev”). и car("Gorbunov", "Audi", "White", 2500000, "Korolev"). Car\_Mark конкретизируется с “Audi”, Car\_Cost – с 2500000, сравнивается фамилия владельца и город.  Результат – car(“Gorbunov”, “Audi”, \_, 2500000, “Korolev”). | Прямой ход, производится подстановка значений, которые были найдены в исходный вопрос. При этом Phone = “89251472838”, Surname = “Gorbunov”, City = “Korolev”, Car\_Mark = “Audi”, Car\_Cost = 2500000. |
| 4 | find(“89251472838”, “Audi”, 2500000) |  |

1. find("89251472838", Car\_Mark) – ненужные промежуточные шаги опущены.

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; результат; подстановка, если есть | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
| 1 | Производится сравнение find(“89251472838”, Car\_Mark). и find(Phone, Car\_Mark). Phone конкретизируется с “89251472838”. Связываются Car\_Mark с Car\_Mark.  Результат – find(“89251472838”, Car\_Mark) | Прямой ход, производится переход к унификации find(Phone, Car\_Mark, \_), где переменная Phone равна “89251472838”. |
|  | … | … |
| 2 | Производится сравнение find(“89251472838”, Car\_Mark, \_). и find(Phone, Car\_Mark, Car\_Cost). Phone конкретизируется с “89251472838”. Связываются Car\_Mark с Car\_Mark.  Результат – find(“89251472838”, Car\_Mark, \_) | Прямой ход, производится переход к унификации abonement(Surname, Phone, address(City, \_, \_, \_)), где переменная Phone равна “89251472838”. |
|  | … | … |
| 3 | Производится сравнение abonement(Surname, “89251472838”, address(City, \_, \_, \_)). и ("Gorbunov", "89251472838", address("Korolev", "Glavnaya", 55, 122)). Surname конкретизируется с “Gorbunov”, City – с “Korolev”, сравниваются номера.  Результат – abonement("Gorbunov", "89251472838", address("Korolev", \_, \_, \_)). | Прямой ход, производится переход к унификации car(Surname, Car\_Mark, \_, Car\_Cost, City). При этом Phone = “89251472838”, Surname = “Gorbunov”, City = “Korolev”. |
|  | … | … |
| 4 | Производится сравнение car(“Gorbunov”, Car\_Mark, \_, \_, “Korolev”). и car("Gorbunov", "Audi", "White", 2500000, "Korolev"). Car\_Mark конкретизируется с “Audi”, сравнивается фамилия владельца и город.  Результат – car(“Gorbunov”, “Audi”, \_, \_, “Korolev”). | Прямой ход, производится подстановка значений, которые были найдены в исходный вопрос. При этом Phone = “89251472838”, Surname = “Gorbunov”, City = “Korolev”, Car\_Mark = “Audi”. |
| 5 | find(“89251472838”, “Audi”) |  |

1. City = "Korolev",

Surname = "Gorbunov",

abonement(Surname, Phone, address(City, Street, \_, \_)), deposit(Surname, Bank\_Name, \_, \_, City) – ненужные промежуточные шаги опущены.

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; результат; подстановка, если есть | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
| 1 | Производится сравнение abonement(“Gorbunov”, Phone, address(“Korolev”, Street, \_, \_)). и abonement("Gorbunov", "89251472838", address("Korolev", "Glavnaya", 55, 122)). Phone конкретизируется с “89251472838”, Street – с “Glavnaya”.  Результат – abonement(“Gorbunov”, “89251472838”, address(“Korolev”, “Glavnaya”, \_, \_)) | Прямой ход, производится переход к унификации deposit(Surname, Bank\_Name, \_, \_, City). При этом переменная Surname равна “Gorbunov”, City равна “Korolev”, Phone – “89251472838”, Street – “Glavnaya”. |
|  | … | … |
| 2 | Производится сравнение deposit(“Gorbunov”, Bank\_Name, \_, \_, “Korolev”). и deposit("Gorbunov", "Sberbank", "123456789", 30000, "Korolev"). Bank\_Name конкретизируется с “Sberbank”.  Результат – deposit(“Gorbunov”, “Sberbank”, \_, \_, “Korolev”) | Прямой ход, производится подстановка значений, которые были найдены в исходный вопрос. При этом Surname = “Gorbunov”, Bank\_Name = “Sberbank”, Phone = “89251472838”, City = “Korolev”, Street = “Glavnaya”. |
| 3 | abonement(“Gorbunov”, “89251472838”, address(“Korolev”, “Glavnaya”, \_, \_)), deposit(“Gorbunov”, “Sberbank”, \_, \_, “Korolev”) |  |

**Ответы на вопросы**

1. **Что такое терм?**

Терм – основной элемент языка Prolog. Терм может быть:

* Константой
  + Число
  + Символьный атом (комбинация символов латинского алфавита, цифр и символа подчеркивания)
  + Строка
* Переменной
  + Именнованная
  + Анонимная (обозначается символом подчеркивания)
* Составной терм (средство организации групп отдельных элементов знаний в единый объект).

1. **Что такое предикат в матлогике (математике)?**

Предикат – высказывание, содержащее одну или несколько неизвестных переменных. Высказывание – это предложение, о котором можно судить, верно оно или нет. До тех пор, пока не будут определены все неизвестные переменные предиката, невозможно сказать истина это, или ложь. Также, предикат – функция со множеством значений {0, 1} или {Ложь, Правда}, определенная на множестве.

1. **Что описывает предикат в Prolog?**

Предикат описывает отношение, определяемое процедурой. Процедура – совокупность правил, заголовки которых одинаковы.

1. **Назовите виды предложений в программе и приведите примеры таких предложений из Вашей программы. Какие предложения являются основными, а какие – не основными? Каковы: синтаксис и семантика (формальный смысл) этих предложений (основных и неосновных)?**

В Prolog существует два основных вида предложений – правила и факты. Правила – предложение вида , то, что находится слева от знака является заголовком правила, то, что справа – телом правила. Факт – частный случай правила, у него нет тела. Также предложения бывают основными и неосновными. Основные – предложения, не содержащие переменные. Неосновные – наоборот, содержащие.

1. **Каковы назначение, виды и особенности использования переменных в программе на Prolog? Какое предложение БЗ сформулировано в более общей – абстрактной форме: содержащее или не содержащее переменных?**

Переменные нужны для обозначения некоторого неизвестного объекта предметной области. Переменные бывают именованными или анонимными. Переменная уникальна в рамках предложения. Предложение, содержащее переменные сформулировано в более общей форме, так как заранее неизвестно значение этой переменной.

1. **Что такое подстановка?**

Применение подстановки заключается в замене каждого вхождения неизвестной переменной предиката на соответствующий терм, который задается в вопросе. Это замена одного терма на другой.

1. **Что такое пример терма? Как и когда строится? Как Вы думаете, система строит и хранит примеры?**

Пример терма – это результат подстановки некоторых конкретных значений в предикат, частный случай предиката. Строится после того, как задан вопрос. Хранится до окончания работы программы.