|  |  |
| --- | --- |
| 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» (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 11/12/13 |
|  |  |

**Дисциплина:** Функциональное и логическое программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-62Б |  |  | Е.В. Брянская |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Н.Б.Толпинская  Ю.В.Строганов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2021

**Л/р 11**

**Задание**

Составить программу, с помощью которой можно определить, например, множество студентов, обучающихся в одном вузе.

|  |
| --- |
| DOMAINS  name, surname, subj, unv = symbol.    PREDICATES  student(name, surname, unv).  teacher(name, surname, subj, unv).    CLAUSES  student("Ann", "Richy", "unv\_1").  student("Boby", "Adamson", "unv\_3").  student("Galya", "Backer", "unv\_2").  student("Gigi", "Dyson", "unv\_1").    student("Tanya", "Park", Unv):-student("Ann", "Richy", Unv).    teacher("Kristina", "Nikiforova", "physics", "unv\_2").  teacher("Mila", "Kolovanova", "english", "unv\_2").  teacher("Anton", "Popov", "drawing", "unv\_4").    GOAL  %student("Boby", "Adamson", "unv\_3").    %student("Galya", Surname, "unv\_2").    %teacher(\_, \_, \_, "unv\_2").    %teacher(Name, Surname, Subj, "unv\_2").  %teacher(Name, Surname, Subj, "unv\_100"). |

|  |  |  |
| --- | --- | --- |
| 1 | student("Boby", "Adamson", "unv\_3"). |  |
| 2 | student("Galya", Surname, "unv\_2"). |  |
| 3 | teacher(\_, \_, \_, "unv\_2"). |  |
| 4 | teacher(Name, Surname, Subj, "unv\_2"). |  |
| 5 | teacher(Name, Surname, Subj, "unv\_100"). |  |

**Л/р 12**

**Задание**

Составить программу – базу знаний, с помощью которой можно определить, например, множество студентов, обучающихся в одном ВУЗе. Студент может одновременно обучаться в нескольких ВУЗах. Привести примеры возможных вариантов вопросов и варианты ответов (не менее 3-х). Описать порядок формирования вариантов ответа.

|  |
| --- |
| DOMAINS  name, surname, subj, unv = symbol.    PREDICATES  student(name, surname, unv).  teacher(name, surname, subj, unv).    CLAUSES  student("Ann", "Richy", "unv\_1").  student("Boby", "Adamson", "unv\_3").  student("Boby", "Dicson", "unv\_1").  student("Galya", "Backer", "unv\_2").  student("Gigi", "Dyson", "unv\_1").  student("Gigi", "Dyson", "unv\_2").  student("Gigi", "Dyson", "unv\_3").    student("Tanya", "Park", Unv):-student("Ann", "Richy", Unv).    teacher("Kristina", "Nikiforova", "physics", "unv\_2").  teacher("Mila", "Kolovanova", "english", "unv\_2").  teacher("Anton", "Popov", "drawing", "unv\_4").    GOAL  %student(Name, Surname, "unv\_1").    %student("Boby", Surname, "unv\_2").    %teacher(\_, \_, "english", "unv\_2").    %teacher(Name, Surname, Subj, "unv\_4").  %teacher("Mila", "Kolovanova", "english", "unv\_2"). |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | student(Name, Surname, "unv\_1"). | Система сравнивает весь вопрос с первым фактом. В вопросе две переменные, система свяжет их с соответствующими значениями знания (при условии, что природа аргументов одинаковая) и ответит на поставленный вопрос, если ответ да, то в качестве побочного эффекта на экран будут выведены значения переменных. Чтобы продолжить поиск система отменяет это решение (то есть переменная теряет своё значение), и далее продолжается поиск решения. |  |
| 2 | student("Boby", Surname, "unv\_2"). | Аналогично, только в вопросе только одна переменная и две константы. |  |
| 3 | teacher(\_, \_, "english", "unv\_2"). | Аналогично, только разница в том, что не столько важно значение анонимных переменных, сколько важно количество аргументов |  |
| 4 | teacher(Name, Surname, Subj, "unv\_4"). | Аналогично вопросу №1/2 |  |
| 5 | teacher("Mila", "Kolovanova", "english", "unv\_2"). | В вопросе нет переменных, поэтому проверяется только совпадение констант. |  |

**Вопросы**

1. **Что представляет из себя программа на Prolog?**

Программа на Prolog состоит из базы знаний, которую составляют факты и правила.

1. **Структура программы на Prolog?**

Программа на Prolog состоит из разделов, которые начинаются с заголовка.

Разделы:

* Директивы компилятора – зарезервированные символьные константы
* CONSTANTS – раздел описания констант
* DOMAINS – раздел описания доменов
* DATABASE – раздел описания предикатов внутренней базы данных
* PREDICATES – раздел описания предикатов
* CLAUSES – раздел описания предложений базы знаний
* GOAL - раздел описания внутренней цели (вопроса)

В программе могут быть не все разделы.

1. **Как реализуется программа на Prolog?**

Описывается база знаний (факты и правила), задаётся вопрос.

1. **Как формируются результаты работы программы?**

Система пытается найти среди базы знаний такие значения переменных, чтобы ответить «Да» на поставленный вопрос.

**Л/р 13**

**Задание**

Составить программу, то есть 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 т.д.
* «Вкладчики банков»: Фамилия, Банк, счет, сумма, др.

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

Используя правила, обеспечить возможность поиска:

* 1. По № телефона найти: Фамилию, Марку автомобиля, Стоимость автомобиля (может быть несколько).
  2. 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 задания 1 и задания 2:

Для одного из вариантов ответов, и для а и для b, 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.

**Вопросы**

1. **Что такое терм?**

Основной элемент – терм.

Терм:

1. Константа
   1. Число (целое, вещественное)
   2. Символьный атом (комбинация символов латинского алфавита, цифр и \_, **начинающаяся** **со** **строчной буквы**)
   3. Строка (последовательность символов, заключенных в “ “)
2. Переменная
   1. Именованная (комбинация символов латинского алфавита, цифр, **начинающаяся с прописной буквы или с \_**)
   2. Анонимная (обозначается символом \_)
3. Составной терм (средство организации группы отдельных элементов знаний в единый объект). Состоит из функтора (имя отношения) и аргументов, представляющих из себя термы.
4. **Что такое предикат в матлогике (математике)?**

Предикат в матлогике – это логическая функция, возвращающая либо истину, либо ложь.

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

Предикат в Prolog – отношение, определяемое процедурой, утверждение базы знаний.

Процедура – множество предложений базы знаний, которые определяют одно значение, заголовки которых имеют одинаковые функторы, одинаковое количество аргументов одной природы.

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)?**

Факты – частный случай правила, с пустым телом.

Пример: phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)).

Правила – состоят из заголовка (содержится знание) и тела (содержит условие истинности)

Пример: car\_by\_phone(Phone, Surname, Brand, Price) :-

phone\_book(Surname, Phone, \_),

car(Surname, Brand, \_, Price, \_).

Вопрос – состоят только из тела, с их помощью пользователь может задать вопрос системе, и та, используя базу знаний даёт соответствующий ответ.

Пример: brand\_by\_phone("+123456", Brand).

Предложения, не содержащие переменные, называются основными, и, наоборот, предложения, использующие переменные, называются неосновными.

Синтаксис:

<заголовок> :- <тело правила>.

Семантика основных предложений заключается в том, что формируется безусловная истина, неосновных – условная.

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

Переменные нужны для обобщения, в программе нужны как способ передачи значения во времени и пространстве.

**Виды** переменных:

* Именованная (комбинация символов латинского алфавита, цифр, **начинающаяся с прописной буквы или с \_**) (позволяет передавать значения в пространстве и во времени)
* Анонимная (обозначается символом \_) (значение неважно)

Чем больше переменных, тем более общая формулировка знания. Следовательно, предложение, содержащее переменные сформулировано в более абстрактной форме.

1. **Что такое подстановка?**

Подстановкой называется множество пар, вида: {}, где - переменная, – терм. Применение подстановки заключается в замене каждого вхождения переменной на соответствующий терм.

Пусть Θ = { =, … = tn} - подстановка, тогда результат применения подстановки к терму обозначается как АΘ.

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

Терм В называется **примером терма** А, если существует такая подстановка Θ, что В = А Θ, где А Θ – результат применения подстановки к терму. Строятся в процессе работы алгоритма унификации. Система строит примеры в процессе поиска среди базы знаний такие значения переменных, чтобы ответить «Да» на поставленный вопрос, и хранит их до окончания работы программы.

|  |
| --- |
| DOMAINS  surname, phone = symbol.    city, street = symbol.  home, flat = integer.    brand, color = symbol.  price = real.  years = integer.    bank, account = symbol.  sum = real.    address = address(city, street, home, flat).    PREDICATES  phone\_book(surname, phone, address).    car(surname, brand, color, price, years).    investor(surname, bank, account, sum).    car\_by\_phone(phone, surname, brand, price).    brand\_by\_phone(phone, brand).    address\_by\_surname\_city(surname, city, street, bank, phone).  CLAUSES  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)).  phone\_book("Nikiforov", "+987456", address("Zhukovsky", "Gagarin Street", 64, 32)).  phone\_book("Nikiforov", "+111111", address("Zhukovsky", "Gagarin Street", 64, 32)).  phone\_book("Mironova", "+909090", address("Zhukovsky", "Sunny Street", 2, 89)).  phone\_book("Filin", "+444000", address("Ramenskoe", "Central", 1, 1)).  phone\_book("Mironova", "+333333", address("Ramenskoe", "New Street", 77, 77)).    car("Birukova", "BMW", black, 4500000, 2).  car("Birukova", "Ford", white, 6200000, 1).  car("Filin", "Honda", grey, 2300000, 4).    investor("Birukova", "New\_1", deposit, 1000000).  investor("Legneva", "Old\_bank", special, 2370000).  investor("Mironova", "Old\_bank", deposit, 5000).    car\_by\_phone(Phone, Surname, Brand, Price) :-  phone\_book(Surname, Phone, \_),  car(Surname, Brand, \_, Price, \_).    brand\_by\_phone(Phone, Brand) :-  car\_by\_phone(Phone, \_, Brand, \_).    address\_by\_surname\_city(Surname, City, Street, Bank, Phone) :-  phone\_book(Surname, Phone, address(City, Street, \_, \_)),  investor(Surname, Bank, \_, \_).  GOAL  %car\_by\_phone("+123456", Surname, Brand, Price).  %car\_by\_phone("+000000", Surname, Brand, Price).  %car\_by\_phone("+444000", Surname, Brand, Price).    %brand\_by\_phone("+123456", Brand).  %brand\_by\_phone("+444000", Brand).  %brand\_by\_phone("+\*\*\*\*\*\*", Brand).    %address\_by\_surname\_city("Mironova", "Zhukovsky", Street, Bank, Phone).  %address\_by\_surname\_city("Legneva", "Ramenskoe", Street, Bank, Phone). |

**Результаты работы программы**

|  |  |
| --- | --- |
| car\_by\_phone("+123456", Surname, Brand, Price). |  |
| car\_by\_phone("+000000", Surname, Brand, Price). |  |
| car\_by\_phone("+444000", Surname, Brand, Price). |  |
| brand\_by\_phone("+123456", Brand). |  |
| brand\_by\_phone("+444000", Brand). |  |
| brand\_by\_phone("+\*\*\*\*\*\*", Brand). |  |
| address\_by\_surname\_city("Mironova", "Zhukovsky", Street, Bank, Phone). |  |
| address\_by\_surname\_city("Legneva", "Ramenskoe", Street, Bank, Phone). |  |

1. **car\_by\_phone("+123456", Surnamet, Brandt, Pricet).**

|  |  |  |
| --- | --- | --- |
| **№** | **Сравниваемые термы; результат; подстановка, если есть** | **Дальнейшие действия: прямой ход или откат (к чему приводит?)** |
| 0. |  | Начальное состояние резольвенты:  car\_by\_phone("+123456", Surnamet, Brandt, Pricet) |
|  | car\_by\_phone("+123456", Surnamet, Brandt, Pricet)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | car\_by\_phone("+123456", Surnamet, Brandt, Pricet)  =  car("Birukova", "BMW", black, 4500000, 2) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | car\_by\_phone("+123456", Surnamet, Brandt, Pricet)  =  investor("Birukova", "New\_1", deposit, 1000000) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | car\_by\_phone("+123456", Surnamet, Brandt, Pricet)  =  car\_by\_phone(Phone, Surname, Brand, Price)  **Подстановка:**  {Phone = "+123456", Surname = Surnamet, Brand = Brandt, Price = Pricet} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  phone\_book(Surnamet, "+123456", \_)  car(Surnamet, Brandt, \_, Pricet, \_) |
|  | phone\_book(Surnamet, "+123456", \_)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145))  **Подстановка:**  {Phone = "+123456", Surname = Surnamet, Brand = Brandt, Price = Pricet, Surnamet = “Birukova”} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  car("Birukova", Brandt, \_, Pricet, \_) |
|  | car("Birukova", Brandt, \_, Pricet, \_)  =  car("Birukova", "BMW", black, 4500000, 2)  **Подстановка:**  {Phone = "+123456", Surname = Surnamet, Brand = Brandt, Price = Pricet, Surnamet = “Birukova”, Brandt = “BMW”, Pricet = 4500000} | удача  **Изменение резольвенты**  Резольвента пустая  Выводится: Surnamet =“Birukova”, Brandt ="BMW", Pricet =4500000  Откат |
|  | \* \* \* | \* \* \* |

1. **brand\_by\_phone("+123456", Brandt).**

|  |  |  |
| --- | --- | --- |
| **№** | **Сравниваемые термы; результат; подстановка, если есть** | **Дальнейшие действия: прямой ход или откат (к чему приводит?)** |
| 0. |  | Начальное состояние резольвенты:  brand\_by\_phone("+123456", Brandt) |
|  | brand\_by\_phone("+123456", Brandt)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | brand\_by\_phone("+123456", Brandt)  =  car("Birukova", "BMW", black, 4500000, 2) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | brand\_by\_phone("+123456", Brandt)  =  investor("Birukova", "New\_1", deposit, 1000000) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | brand\_by\_phone("+123456", Brandt)  =  car\_by\_phone(Phone, Surname, Brand, Price) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | brand\_by\_phone("+123456", Brandt)  =  brand\_by\_phone(Phone, Brand)  **Подстановка:**  {Phone = “+123456”, Brand = Brandt} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  car\_by\_phone(“+132456”, \_, Brandt, \_) |
|  | car\_by\_phone(“+132456”, \_, Brandt, \_)  и  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | car\_by\_phone(“+132456”, \_, Brandt, \_)  =  car\_by\_phone(Phone, Surname, Brand, Price)  **Подстановка:**  {Phone = "+123456", Brand = Brandt, Phone = “+123456”, Brand = Brandt} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  phone\_book(Surname, "+123456", \_)  car(Surname, Brandt, \_, Price, \_) |
|  | phone\_book(Surname, "+123456", \_)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145))  **Подстановка:**  {Phone = "+123456", Brand = Brandt, Phone = “+123456”, Brand = Brandt , Surname = “Birukova”} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  car("Birukova", Brandt, \_, Price, \_) |
|  | car("Birukova", Brandt, \_, Price, \_)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | car("Birukova", Brandt, \_, Price, \_)  =  car("Birukova", "BMW", black, 4500000, 2)  **Подстановка:**  {Phone = "+123456", Brand = Brandt, Phone = “+123456”, Brand = Brandt, Surname = “Birukova”, Brandt = “BMW”, Price = 4500000} | удача  **Изменение резольвенты**  Резольвента пустая  Выводится: Brandt ="BMW"  Откат |
|  | \* \* \* | \* \* \* |

1. **address\_by\_surname\_city("Mironova", "Zhukovsky", Streett, Bankt, Phonet).**

|  |  |  |
| --- | --- | --- |
| **№** | **Сравниваемые термы; результат; подстановка, если есть** | **Дальнейшие действия: прямой ход или откат (к чему приводит?)** |
| 0. |  | Начальное состояние резольвенты:  address\_by\_surname\_city("Mironova", "Zhukovsky", Streett, Bankt, Phonet) |
|  | address\_by\_surname\_city("Mironova", "Zhukovsky", Streett, Bankt, Phonet)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | address\_by\_surname\_city("Mironova", "Zhukovsky", Streett, Bankt, Phonet)  =  address\_by\_surname\_city(Surname, City, Street, Bank, Phone)  **Подстановка:**  {Surname = “Mironova”, City = “Zhukovsky”, Street=Streett, Bank = Bankt, Phone = Phonet} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  phone\_book(“Mironova”, Phonet, address(“Zhukovsky”, Streett, \_, \_)),  investor(“Mironova”, Bankt, \_, \_) |
|  | phone\_book(“Mironova”, Phonet, address(“Zhukovsky”, Streett, \_, \_))  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | phone\_book(“Mironova”, Phonet, address(“Zhukovsky”, Streett, \_, \_))  =  phone\_book("Mironova", "+909090", address("Zhukovsky", "Sunny Street", 2, 89))  **Подстановка:**  {Surname = “Mironova”, City = “Zhukovsky”, Street=Streett, Bank = Bankt, Phone = Phonet , Phonet = “+909090”, Streett = "Sunny Street"} | удача  **Изменение резольвенты**  Новое состояние резольвенты:  investor(“Mironova”, Bankt, \_, \_) |
|  | investor(“Mironova”, Bankt, \_, \_)  =  phone\_book("Birukova", "+123456", address("Moscow", "Zhukovsky Street", 12, 145)) | неудача  (не совпали функторы) |
|  | \* \* \* | \* \* \* |
|  | investor(“Mironova”, Bankt, \_, \_)  =  investor("Mironova", "Old\_bank", deposit, 5000)  **Подстановка:**  {Surname = “Mironova”, City = “Zhukovsky”, Street=Streett, Bank = Bankt, Phone = Phonet , Phonet = “+909090”, Streett = "Sunny Street", Bankt=”Old\_bank”} | удача  **Изменение резольвенты**  Резольвента пустая  Выводится: Streett = "Sunny Street", Brankt ="Old\_bank", Phonet = “+909090”  Откат |
|  | \* \* \* | \* \* \* |