|  |  |
| --- | --- |
| 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**

**по лабораторной работе № 1**

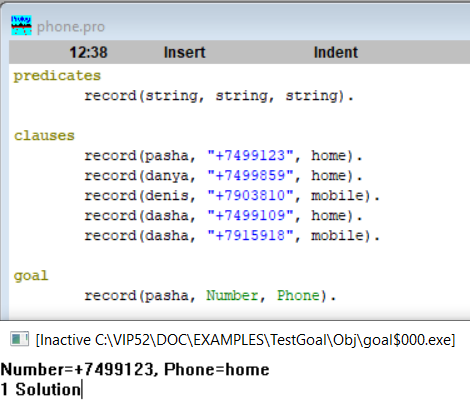
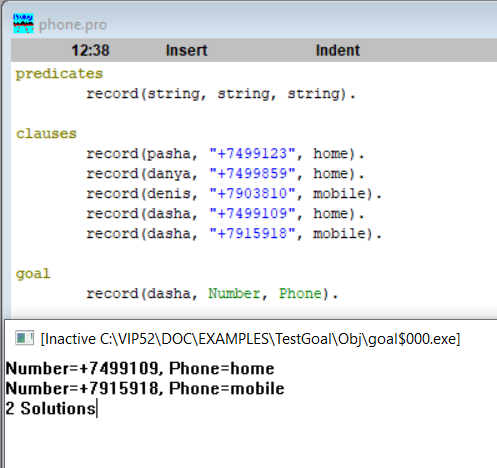
|  |  |
| --- | --- |
| **Дисциплина:** Функциональное и логическое программирование  **Студент:** Платонова Ольга  **Группа:** ИУ7-65Б  **Преподаватели:** Толпинская Н. Б.  Строганов Ю. В. |  |

Москва, 2021 г.

**Часть 1.**

Запустить среду Visual Prolog 5.2. Настроить утилиту TestGoal. Запустить тестовую программу, проанализировать реакцию системы и множество ответов. Разработать свою программу – «Телефонный справочник». Протестировать работу программы.

*Листинг 1.*

*Вывод 1.*

*Что собой представляет программа на Prolog?*

Программа на Prolog представляет собой базу знаний и вопрос. База знаний состоит из предложений – CLAUSES: фактов и правил. Каждое предложение заканчивается точкой. Вопрос является составным термом. Система рассматривает вопрос как цель, к которой надо стремиться.

*Какова структура программы?*

Программа на Prolog состоит из разделов, который начинается со своего заголовка.

* 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.

*Как программа реализуется, как формируются результаты работы программы?*

В процессе выполнения программы – система, используя встроенный алгоритм унификации, пытается обосновать возможность истинности вопрос, строя подстановки и примеры термов. Алгоритм унификации автоматически и многократно запускается системой.

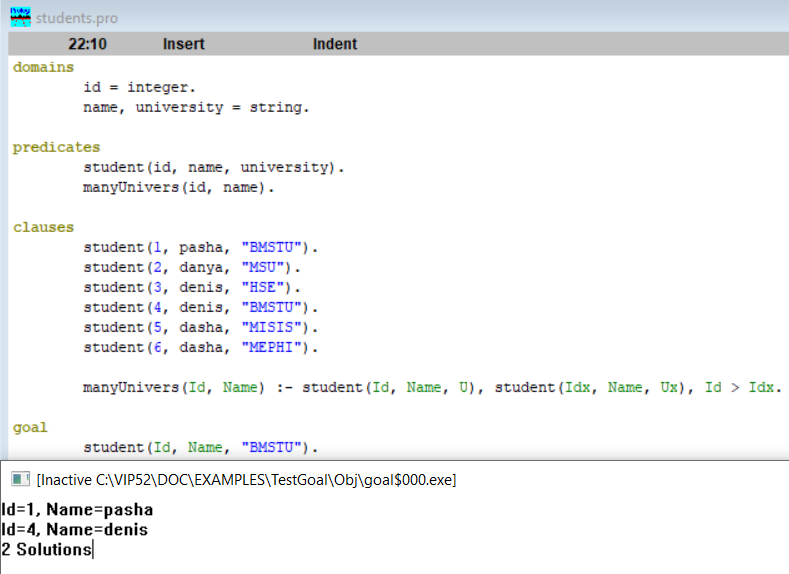
При запуске алгоритма унифицирования в стек помещается равенство B = AƟ. Затем из стека считывается рабочее поле и обрабатывается 1 равенство. Цикл завершается, если стек пуст или поступило сообщение об отказе. Дополнительно используется переменная «неудача». В результате работы алгоритма унификации результирующая подстановка будет располагаться в результирующей ячейке памяти.

**Часть 2.**

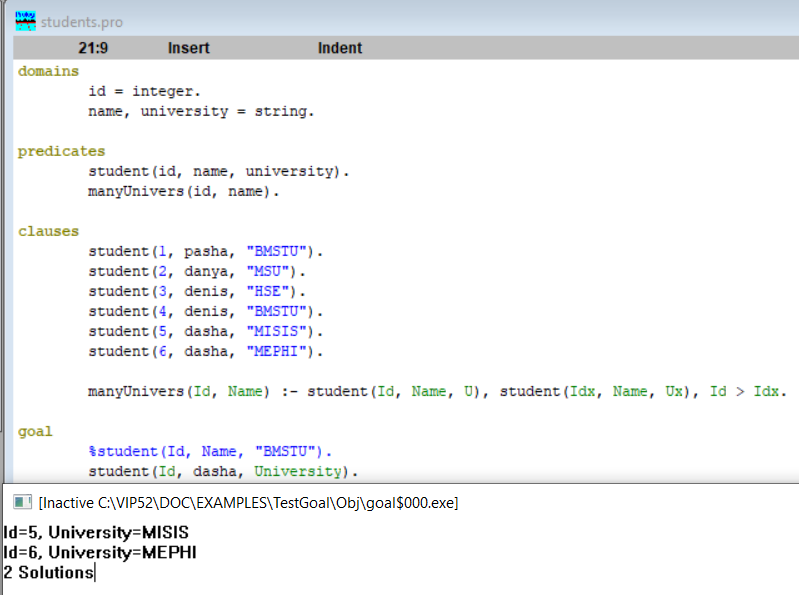
Составить программу – базу знаний, с помощью которой можно определить, например, множество студентов, обучающихся в одном ВУЗе. Студент может одновременно обучаться в нескольких ВУЗах. Привести примеры возможных вариантов вопросов и варианты ответов. Описать порядок формирования ответа. Исходную базу знаний сформировать с помощью только фактов.

*Листинг 2.*

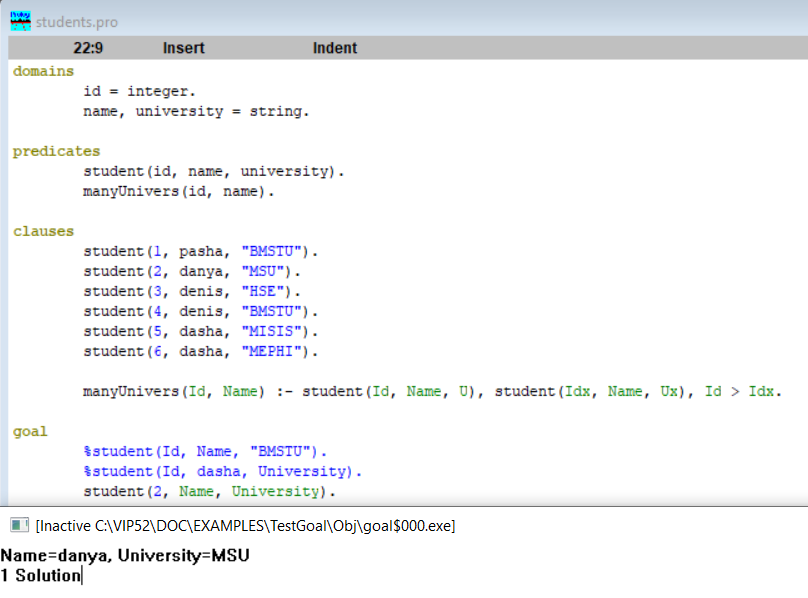
1. Множество студентов из университета “BMSTU”.

**

1. Множество университетов, в которых обучается “dasha”.

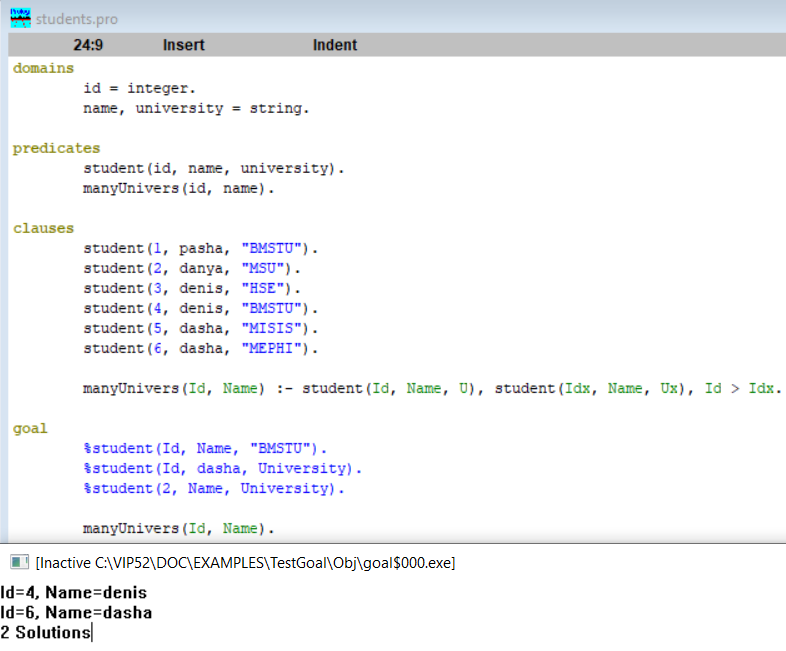


1. Информация о студенте номер 2.



1. Список студентов, обучающихся более, чем в 1 ВУЗе.

В данном примере используется правило “manyUniverse”.



*Вывод 2.*

*Назначение и использование переменных.*

Факты содержат переменные (Id, Name, University), которые записываются с большой буквы или с \_. Вообще – переменные предназначены для передачи знаний «во времени и в пространстве». Переменная – способ обобщить, входит в факт с квантором всеобщности (т.е. любой элемент из множества).

*Что собой представляет программа на Prolog, какова ее структура, как она реализуется.*

Программа на Prolog представляет собой базу знаний и вопрос. База знаний состоит из предложений – CLAUSES: фактов и правил. Каждое предложение заканчивается точкой. Вопрос является составным термом. Система рассматривает вопрос как цель, к которой надо стремиться.

Программа на Prolog состоит из разделов, который начинается со своего заголовка.

* 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.

В процессе выполнения программы – система, используя встроенный алгоритм унификации, пытается обосновать возможность истинности вопрос, строя подстановки и примеры термов. Алгоритм унификации автоматически и многократно запускается системой.

При запуске алгоритма унифицирования в стек помещается равенство B = AƟ. Затем из стека считывается рабочее поле и обрабатывается 1 равенство. Цикл завершается, если стек пуст или поступило сообщение об отказе. Дополнительно используется переменная «неудача». В результате работы алгоритма унификации результирующая подстановка будет располагаться в результирующей ячейке памяти.

**Часть 3.**

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

* «Телефонный справочник»: фамилия, телефон, адрес – структура (город, улица, № дома, № квартиры),
* Автомобили: фамилия, марка, цвет, стоимость и др.,
* Вкладчики банков: фамилия, банк, счет, сумма, др.

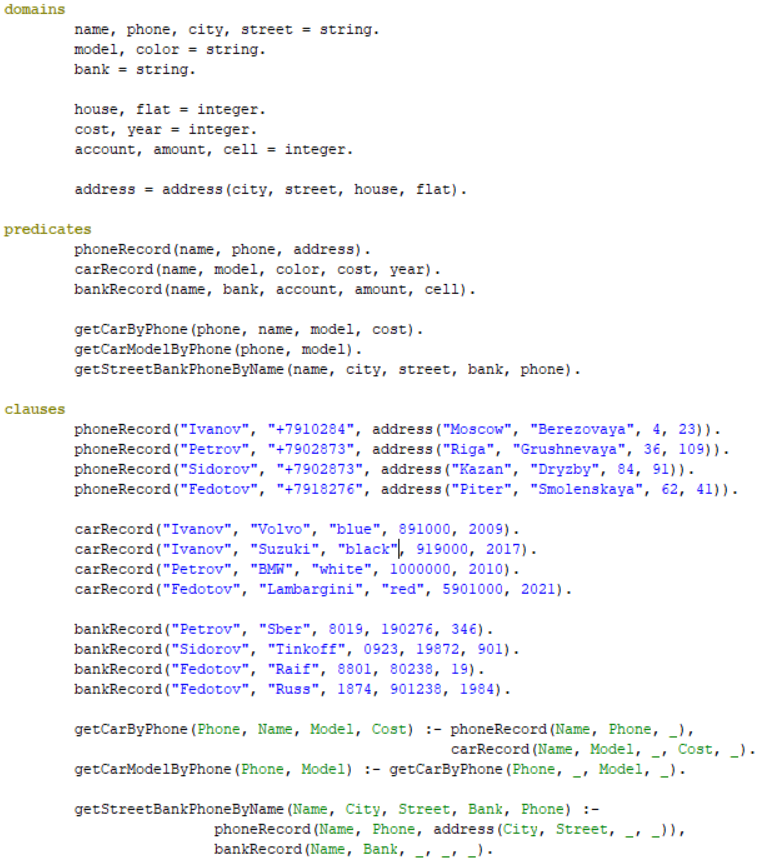
Владелец может иметь несколько телефонов, автомобилей, вкладов. Используя правила, обеспечить возможность поиска:

1. А) По № телефона найти: фамилию, марку автомобиля, стоимость автомобиля

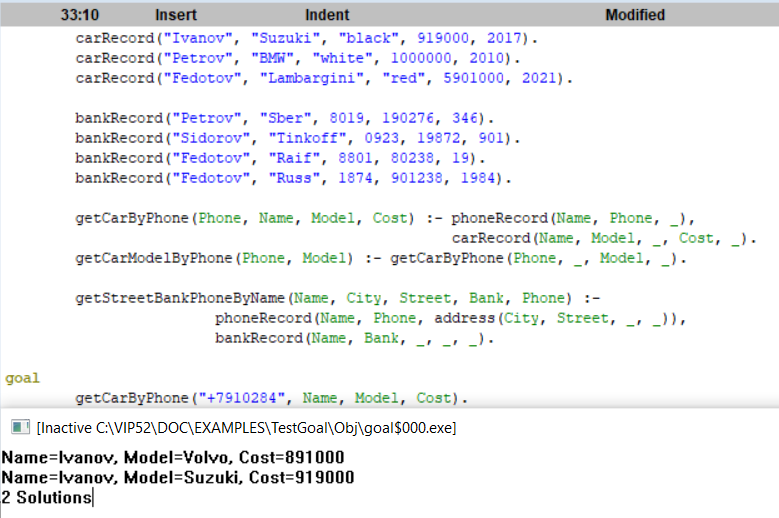
Б) Используя правило а), по № телефона найти марку автомобиля.

1. Используя простой вопрос по фамилии и городу найти: улицу, банки и № телефона.

*Листинг 3.*



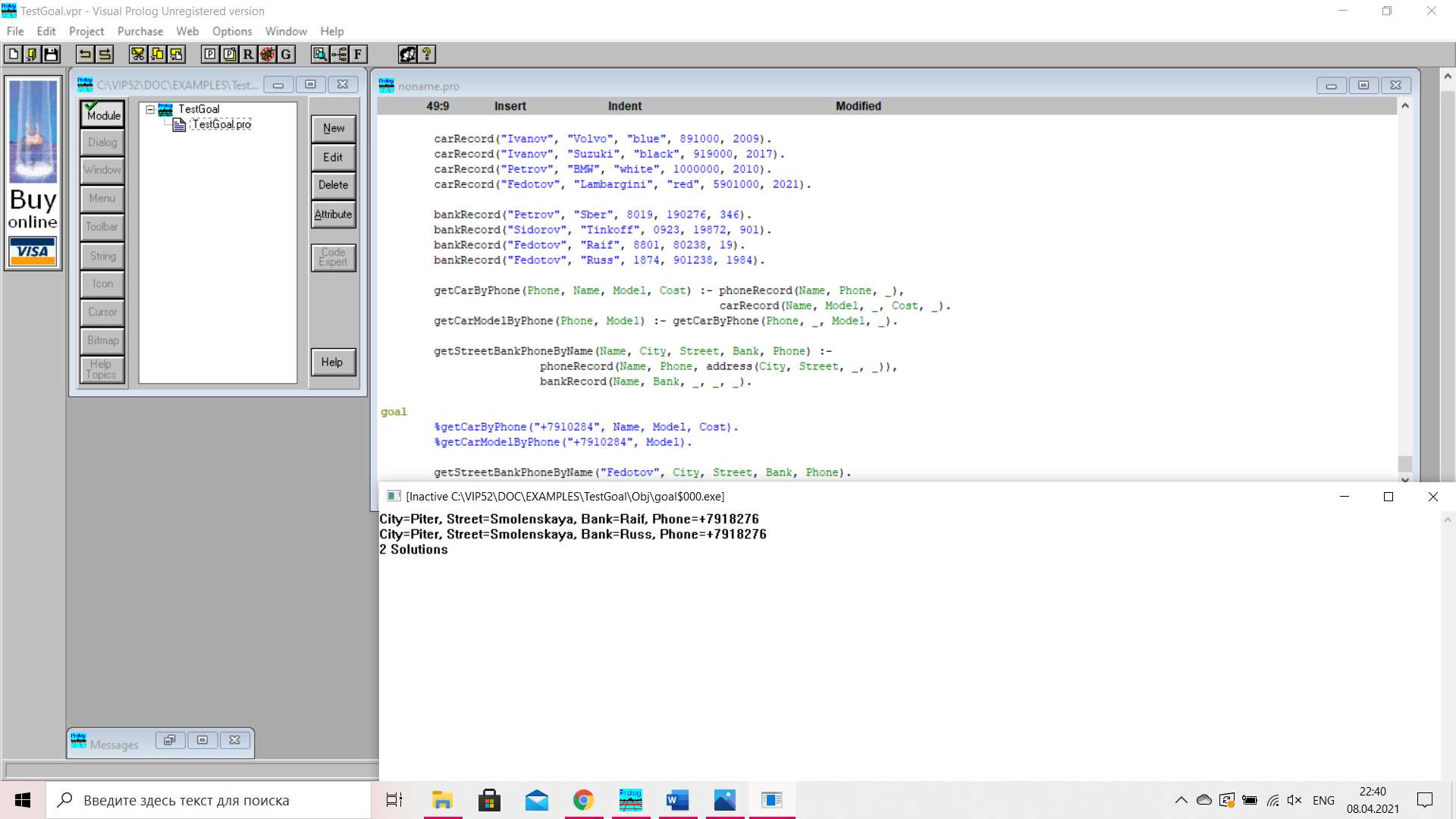
1. По № телефона найти: фамилию, марку автомобиля, стоимость автомобиля.



1. Используя правило а), по № телефона найти марку автомобиля.



1. Используя простой вопрос по фамилии и городу найти: улицу, банки и № телефона.



*Вывод 3.*

1. *Что такое терм?*

Терм – основной элемент языка.

* 1. Константа
     1. Число (целое, вещественное),
     2. Символьный атом (комбинация букв, цифр и \_),
     3. Строка (последовательность символов в кавычках).
  2. Переменная
     1. Именованная (начинается с заглавной буквы или \_),
     2. Анонимная (символ \_).
  3. Составной терм (f(t1, …, tm), где f – функтор, t1, …, tm - термы).

1. *Что такое предикат в матлогике?*

Предикат – это высказывание, в которое можно подставлять аргументы 0 (ложь) или 1 (истина).

1. *Что описывает предикат в Prolog?*

Предикат в Prolog – утверждение программы. Описывает отношение сущностей базы знаний.

1. *Назовите виды предложения в программе и приведите примеры таких предложений вашей программы.*

Предложение имеет вид: A :- B1, …, Bn. A – заголовок, B1, …, Bn – тело. Предложения бывают двух видов: факты (частный случай правила, тело пустое) и правила (предложения более общего вида).

Пример факта: phoneRecord("Ivanov", "+7910284", address("Moscow", "Berezovaya", 4, 23)).

Пример правила: getCarModelByPhone(Phone, Model) :- getCarByPhone(Phone, \_, Model, \_).

*Какие предложения являются основными, а какие неосновными? Каковы синтаксис и семантика этих предложений?*

Основными называются предложения, которые не содержат переменных. Неосновные в момент фиксации в программе могут содержать переменные.

1. *Каковы назначение, виды и особенности использования переменных в программе на Prolog?*

Вообще – переменные предназначены для передачи значений «во времени и пространстве». Переменные обозначают некоторый неизвестный объект из некоторого множества объектов. Переменные в факты и правила входят только с квантором всеобщности, в вопросы – только с квантором существования. Факты содержат переменные, которые записываются с большой буквы или с \_. Переменная – способ обобщить, входит в факт с квантором всеобщности (т.е. любой элемент из множества).

*Какое предложение БЗ сформулировано в более общей – абстрактной форме: содержащее или не содержащее переменных?*

Содержащее переменные, т.к. чем больше переменных, тем более общая форма.

1. *Что такое подстановка?*

Подстановкой называется множество пар, вида {Xi = ti}, где Xi – переменная, а ti – терм. Применение подстановки заключается в замене каждого вхождения переменной Xi на соответствующий терм.

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

Терм B называется примером терма А, если существует такая подстановка Ɵ, что В = ƟА. Строится во время работы алгоритма унификации. Пример терма хранится до окончания работы программы.

1. а) **getCarByPhone("+7910284", Name, Model, Cost).**

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; результат; подстановка | Дальнейшие действия |
| 1 | getCarByPhone("+7910284", Name, Model, Cost).  getCarByPhone(phone, Name, Model, Cost).  Связываются: Phone = “+7910284” | getCarByPhone() – удаляется из стека  phoneRecord(Name, Phone, \_), carRecord(Name, Model, \_, Cost, \_) – помещаются в стек. |
| 2 | phoneRecord(Name, “+7910284”, \_).  phoneRecord("Ivanov", "+7910284", address("Moscow", "Berezovaya", 4, 23)).  Связываются: Name = “Ivanov” | phoneRecord(Name, Phone, \_) – удаляется из стека |
| 3 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Ivanov", "Volvo", "blue", 891000, 2009).  Связываются: Model = “Volvo”, Cost = 8910000 | carRecord(Name, Model, \_, Cost, \_) – удаляется из стека.  Выводятся «Ivanov», «Volvo», 8910000  carRecord(Name, Model, \_, Cost, \_) – помещается в стек. |
| 4 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Ivanov", "Suzuki", "black", 919000, 2017).  Связываются: Model = “Suzuki”, Cost = 919000 | carRecord(Name, Model, \_, Cost, \_) – удаляется из стека.  Выводятся «Ivanov», «Suzuki», 919000  carRecord(Name, Model, \_, Cost, \_) – помещается в стек. |
| 5 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Petrov", "BMW", "white", 1000000, 2010).  Не унифицируемые. |  |
| 6 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Fedotov", "Lambargini", "red", 5901000, 2021).  Не унифицируемые. | Откат.  Все факты carRecord пройдены. В стек помещается последний вопрос phoneRecord(Name, Phone, \_).  Name развязывается. |
| 7 | Остальные факты phoneRecord не унифицируемые. | Откат.  Все факты phoneRecord пройдены. Последний вопрос getCarByPhone() – помещается в стек.  Phone развязывается. |

1. b) **getCarModelByPhone("+7910284", Model).**

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; результат; подстановка | Дальнейшие действия |
| 1 | getCarModelByPhone("+7910284", Model).  getCarModelByPhone(Phone, Model)  Связываются: Phone = “+7910284” | getCarModelByPhone() – удаляется из стека  getCarByPhone(Phone, \_, Model, \_) - помещается в стек. |
| 2 | getCarByPhone("+7910284", Name, Model, Cost).  getCarByPhone(phone, Name, Model, Cost).  Связываются: Phone = “+7910284” | phoneRecord(Name, Phone, \_) – удаляется из стека |
| 3 | phoneRecord(Name, “+7910284”, \_).  phoneRecord("Ivanov", "+7910284", address("Moscow", "Berezovaya", 4, 23)).  Связываются: Name = “Ivanov” | phoneRecord(Name, Phone, \_) – удаляется из стека |
| 4 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Ivanov", "Volvo", "blue", 891000, 2009).  Связываются: Model = “Volvo”, Cost = 8910000 | carRecord(Name, Model, \_, Cost, \_) – удаляется из стека.  Выводится «Volvo»  carRecord(Name, Model, \_, Cost, \_) – помещается в стек. |
| 5 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Ivanov", "Suzuki", "black", 919000, 2017).  Связываются: Model = “Suzuki”, Cost = 919000 | carRecord(Name, Model, \_, Cost, \_) – удаляется из стека.  Выводятся «Suzuki»  carRecord(Name, Model, \_, Cost, \_) – помещается в стек. |
| 6 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Petrov", "BMW", "white", 1000000, 2010).  Не унифицируемые. |  |
| 7 | carRecord(“Ivanov”, Model, \_, Cost, \_).  carRecord("Fedotov", "Lambargini", "red", 5901000, 2021).  Не унифицируемые. | Откат.  Все факты carRecord пройдены. В стек помещается последний вопрос phoneRecord(Name, Phone, \_).  Name развязывается. |
| 8 | Остальные факты phoneRecord не унифицируемые. | Откат.  Все факты phoneRecord пройдены. Последний вопрос getCarByPhone() – помещается в стек.  Phone развязывается. |
| 9 |  | Откат.  getCarByPhone – удаляется из стека  Последний вопрос getCarModelByPhone (Phone, \_, Model, \_) - помещается в стек. |

2. **getStreetBankPhoneByName("Fedotov", City, Street, Bank, Phone).**

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; результат; подстановка | Дальнейшие действия |
| 1 | getStreetBankPhoneByName("Fedotov", City, Street, Bank, Phone).  getStreetBankPhoneByName(Name, City, Street, Bank, Phone).  Связываются: Name = “Fedotov” | getStreetBankPhoneByName () – удаляется из стека  phoneRecord(Name, Phone, address(City, Street, \_, \_)),  bankRecord(Name, Bank, \_, \_, \_) – помещаются в стек. |
| 2 | phoneRecord(“Fedotov”, Phone, address(City, Street, \_, \_)),  phoneRecord("Fedotov", "+7918276", address("Piter", "Smolenskaya", 62, 41)).  Связываются: Phone = “+7918276”, City = “Piter”, Street = “Smolenskaya” | phoneRecord(Name, Phone, address(City, Street, \_, \_)) – удаляется из стека. |
| 3 | bankRecord(“Fedotov”, Bank, \_, \_, \_).  bankRecord("Fedotov", "Raif", 8801, 80238, 19).  Связываются: Bank = “Raif” | bankRecord(Name, Bank, \_, \_, \_) – удаляется из стека.  Выводятся “Piter”, “Smolenskaya”, “Raif”, “+7918276”  bankRecord(Name, Bank, \_, \_, \_) – помещается в стек. |
| 4 | bankRecord(“Fedotov”, Bank, \_, \_, \_).  bankRecord("Fedotov", "Russ", 1874, 901238, 1984).  Связываются: Bank = “Russ” | bankRecord(Name, Bank, \_, \_, \_) – удаляется из стека.  Выводятся “Piter”, “Smolenskaya”, “Russ”, “+7918276” |
| 5 |  | Откат.  Все факты bankRecord пройдены. В стек помещается последний вопрос phoneRecord(Name, Phone, address(City, Street, \_, \_)). |
| 6 | Остальные факты phoneRecord не унифицируемые. | Откат.  Все факты phoneRecord пройдены. Последний вопрос getStreetBankPhoneByName() – помещается в стек.  Name развязывается. |