|  |  |
| --- | --- |
|  | **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»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ДИСЦИПЛИНА «Функциональное и логическое программирование»

**Лабораторные работы № 11, 12, 13**

**“Структура программы на Prolog”**

|  |  |
| --- | --- |
| **Студент**  Чалый А. А.  **Группа** ИУ7 – 62 Б  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель** Толпинская Н.Б. |  |

Москва.

2021 г.

**Цель работы:**

Познакомиться со структурой, принципами оформления и логикой выполнения программы на Prolog.

**Задачи работы:**

Приобрести навыки декларативного описания предметной области с использованием фактов и правил. Изучить способы использования фактов и правил в программе на Prolog-е, принципы и правила сопоставления и отождествления, принцип унификации.

**Задача 1. Настроить среду Visual Prolog.**

1. **Запускаем среду визуальной разработки Visual Prolog**
2. **Создаем новый проект с помощью “New Project”**
3. **Даем название нашему проекту “TestGoal”**
4. **Во вкладке “Target” устанавливаем необходимые параметры**
5. **Настройка компилятора во вкладке “Compiler options”**
6. **Создать файл test.pro**
7. **Проверим правильность настройки системой вызовом “Test Goal” программы “Hello world”**



Как видно на рисунке программа работает корректно.

**Задание 2.**

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

**Студент может одновременно обучаться в некскольких ВУЗах. Привести примеры возможных вариантов вопросов и варианты ответов (не менее 3-х). Описать порядок формирования вариантов ответа.**

* **Исходную базу знаний сформировать, используя правила.**
* **Разработать свою базу знаний (содержание произвольно).**

Листинг 1. Программа student.pro

domains

name, surname, group, university = symbol.

predicates

study(name, surname, group, university).

getUnisOfStudent(Name, Surname, University)

clauses

study("Andrey", "Chalyy", "IU7-62B", "BMSTU").

study("Nikita", "Gag", "IU7-62B", "BMSTU").

study("D", "Kul", "IU7-62B", "BMSTU").

study("Ilya", "Balachiy", "FFO-32", "TGNK").

study("Maxim", "Drevo", "FIN-11", "MGGU").

study("D", "Kul", "IO-23", "MSU").

getUnisOfStudent (N, S, U) :- study(N, S, \_, U).

GOAL

%study(Name, Surname, Group, "BMSTU"), nl.

%study("D", "Kul", Group, University), nl.

getUnisOfStudent("D", "Kul", University).

В базе знаний содержится информация о студентах: их имя, фамилия, группа и университет.

name — используется для объявления имени

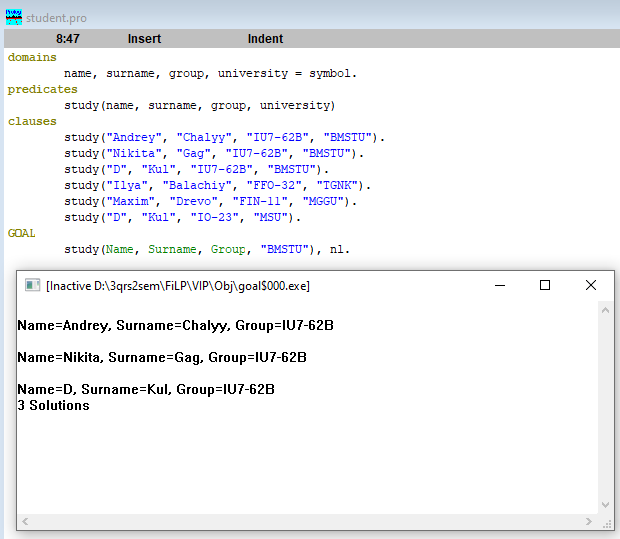
surname — используется для объявления фамилии

group — используется для объявления группы обучения

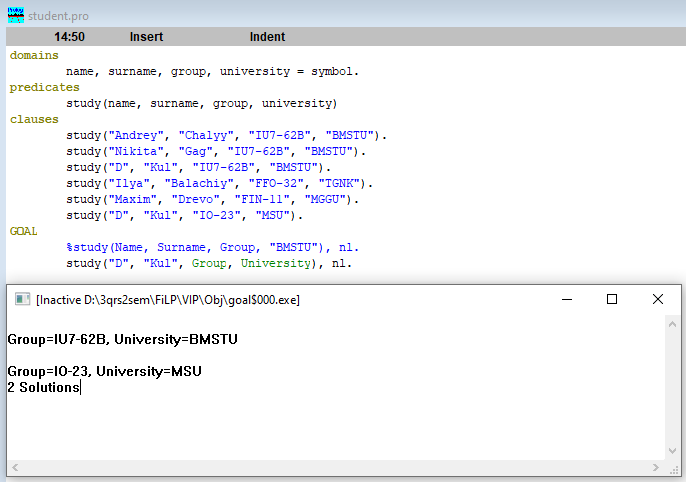
university — используется для объявления университета

**Примеры:**

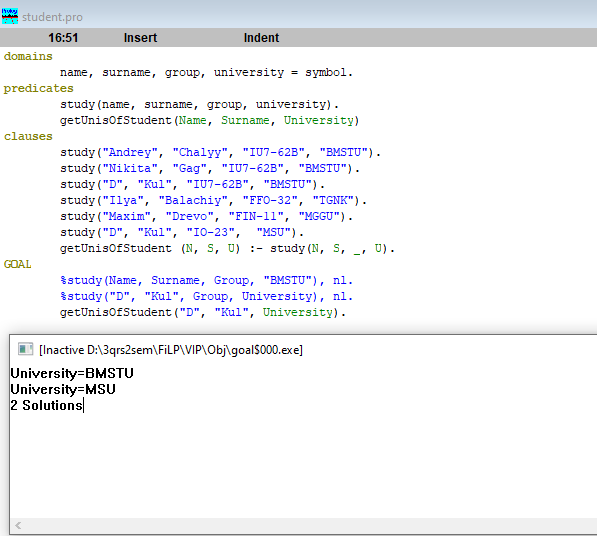
1. **Все студенты обучающиеся в BMSTU**



1. **Вузы и группа в которых учится студент**



1. **Использование правила**



**Задание 3.**

**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),**

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

**2. 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**

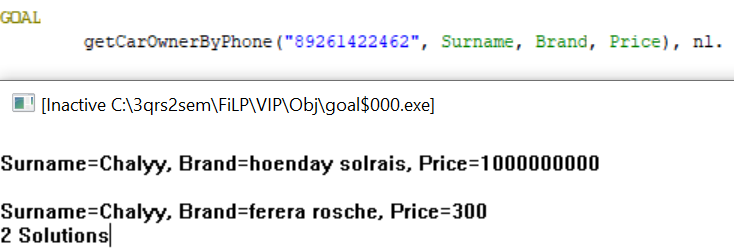
**вклады и № телефона.**

**Для задания 1 и задания 2:**

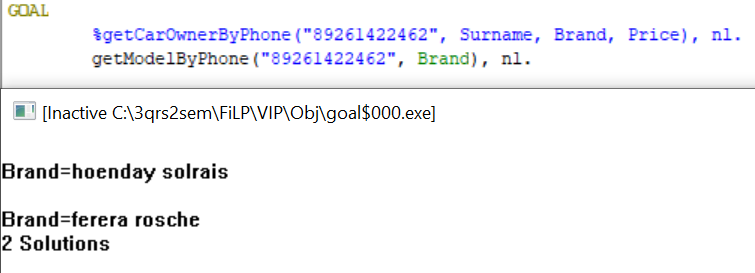
**Для одного из вариантов ответов, и для а) и для в), описать словесно порядок поиска ответа на вопрос, указав, как выбираются знания, и, при этом, для каждого этапа унификации, выписать подстановку — наибольший общий унификатор, и соответствующие примеры термов.**

**Примеры:**

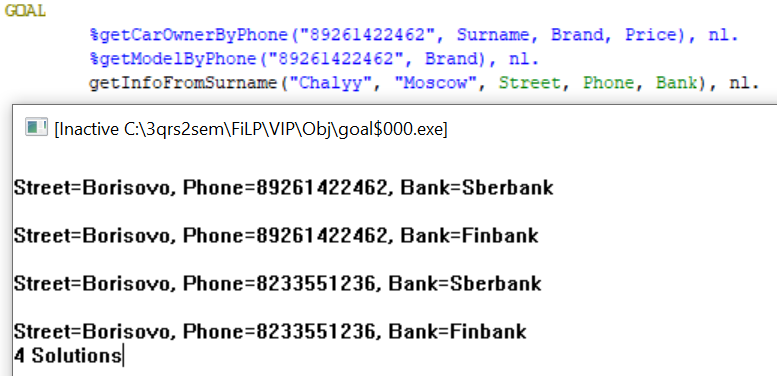
1. **Получить модель и владельца по номеру телефона**



1. **Получить модели по номеру телефона**



1. **Получить информацию по фамилии**



**1.**  getCarOwnerByPhone("89261422462", Surname, Brand, Price).

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; подстановка, если есть | Действия |
| 0. | getCarOwnerByPhone("89261422462", SurnameT, BrandT, PriceT)=person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Разные функторы  Унификация не успешна | Текущее состояние резольвенты:  getCarOwnerByPhone("89261422462", SurnameT, BrandT, PriceT)  Откат, переход к следующему предложению. |
| ... | ... | ... |
| 4. | getCarOwnerByPhone("89261422462", SurnameT, BrandT, PriceT)=car("Chalyy", "hoenday solrais", "blue", 1000000000)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 9. | getCarOwnerByPhone("89261422462", SurnameT, BrandT, PriceT)=depositor("Chalyy", "Sberbank", "812476672", 100210304)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 15. | getCarOwnerByPhone("89261422462", SurnameT, BrandT, PriceT)=getCarOwnerByPhone(Phone, Surname, Brand, Price)  Унификация успешна  = {"89261422462" = Phone, SurnameT = Surname, BrandT = Brand, PriceT = Price} | Новое состояние резольвенты:  person(SurnameT, "89261422462", \_)  car(SurnameT, BrandT, \_, PriceT) |
| 16. | person(SurnameT, "89261422462", \_)=person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Унификация успешна  = {"89261422462" = Phone, SurnameT = Surname, BrandT = Brand, PriceT = Price} | Новое состояние резольвенты:  car("Chalyy", BrandT, \_, PriceT) |
| 17. | car("Chalyy", BrandT, \_, PriceT)=person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению |
| ... | ... | ... |
| 21. | car("Chalyy", BrandT, \_, PriceT)=  car("Chalyy", "hoenday solrais", "blue", 1000000000)  Унификация успешна  = {PhoneT = "89261422462", SurnameT = "Chalyy", BrandT = "hoenday solrais", PriceT = 1000000000} | Вывод:  SurnameT = "Chalyy", BrandT = "hoenday solrais", PriceT = 1000000000  Откат переход к следующему предложению |
| 22. | car("Chalyy", BrandT, \_, PriceT)=  car("Chalyy", "ferera rosche", "red ", 300 )  Унификация успешна  = {PhoneT = "89261422462", SurnameT = "Chalyy", BrandT = "ferera rosche", PriceT = 300 } | Вывод:  SurnameT = "Chalyy", BrandT = "ferera rosche", PriceT = 300  Откат переход к следующему предложению |
| 23. | car("Chalyy", BrandT, \_, PriceT)=  car("DKUL", "Audi A6", "white", 5000000)  Различаются константы surname  Унификация не успешна | Откат переход к следующему предложению |
| ... | ... | ... |
| 26. | car("Chalyy", BrandT, \_, PriceT)=depositor("Chalyy", "Sberbank", "812476672", 100210304)  Разные функторы  Унификация не успешна | Откат переход к следующему предложению |
| ... | ... | ... |
| 32. | car("Chalyy", BrandT, \_, PriceT)=getCarOwnerByPhone(Phone, Surname, Brand, Price)  Разные функторы  Унификация не успешна | Откат переход к следующему предложению |
| 33. | car("Chalyy", BrandT, \_, PriceT)=getModelByPhone(Phone, Brand)  Разные функторы  Унификация не успешна | Откат переход к следующему предложению |
| 34. | car("Chalyy", BrandT, \_, PriceT)=getInfoFromSurname(Surname, City, Street, Phone, Bank)  Разные функторы  Унификация не успешна | Текущее состояние резольвенты:  person(SurnameT, "89261422462", \_)  car(SurnameT, BrandT, \_, PriceT) |
| 35. | person(SurnameT, “89261422462”, \_ ) =person("Chalyy", "8233551236", address("Moscow", "Borisovo", 25, 120))  Различаются константы типа phone  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 37. | person(SurnameT, “89261422462”, \_ ) = car("Chalyy", "hoenday solrais", "blue", 1000000000)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | … |
| 43. | person(SurnameT, “89261422462”, \_ ) = depositor("Chalyy", "Sberbank", "812476672", 100210304)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 44. | person(SurnameT, “89261422462”, \_ )=getCarOwnerByPhone(Phone, Surname, Brand, Price) :- person(Surname, Phone, \_),  car(Surname, Brand, \_, Price)  Разные функторы  Унификация не успешна | Откат переход к следующему предложению |
| 49 | person(SurnameT, “89261422462”, \_ )=getModelByPhone(Phone, Brand)  Разные функторы  Унификация не успешна | Откат переход к следующему предложению |
| 50 | person(SurnameT, “89261422462”, \_ )=getInfoFromSurname(Surname, City, Street, Phone, Bank)  Разные функторы  Унификация не успешна | Откат переход к следующему предложению |
| 51 | getCarOwnerByPhone(“89261422462”, SurnameT, BrandT, PriceT)  = getModelByPhone(Phone, Brand))  Разные функторы  Унификация не успешна | Текущее состояние резольвенты:  carByPhone(“89261422462”, SurnameT, BrandT, PriceT) |
| 52 | getCarOwnerByPhone(“89261422462”, SurnameT, BrandT, PriceT)  = getInfoFromSurname(Surname, City, Street, Phone, Bank)  Разные функторы  Унификация не успешна | Новое состояние резольвенты:  Резольвента пуста |

**2.**  getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)

|  |  |  |
| --- | --- | --- |
| № шага | Сравниваемые термы; подстановка, если есть | Действия |
| 0 | getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)=person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Разные функторы  Унификация не успешна | **Текущее состояние резольвенты:**  getInfoFromSurname(“Kulikov”, “Saransk”, StreetQ, PhoneQ, BankQ)  Откат, переход к следующему предложению. |
| ... | ... | ... |
| 4 | getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)=car("Chalyy", "hoenday solrais", "blue", 1000000000)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 9 | getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)=depositor("Chalyy", "Sberbank", "812476672", 100210304)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 15 | getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)= getCarOwnerByPhone(Phone, Surname, Brand, Price)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| 16 | getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)= getModelByPhone(Phone, Brand) | Откат, переход к следующему предложению. |
| 17 | getInfoFromSurname(“Chalyy”, “Moscow ”, StreetT, PhoneT, BankT)= getInfoFromSurname(Surname, City, Street, Phone, Bank)  Унификация успешна  Θ = {“Chalyy ” = Surname, “Moscow ” = City, StreetT=Street PhoneT = Phone, BankT = Bank} | **Новое состояние резольвенты:**  person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))  depositor(“Chalyy ”, BankT, \_ , \_ ) |
| 18 | person(“Chalyy ”, PhoneT, address(“Moscow ”, StreetT, \_ , \_ ))  =person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Унификация успешна  Θ = {“Chalyy ” = Surname, “Moscow ” = City, StreetT= “Borisovo”, PhoneT = "89261422462", BankT = Bank} | **Новое состояние резольвенты:**  depositor(“Chalyy ”, BankT, \_ , \_ ) |
| 19 | depositor(“Chalyy ”, BankT, \_ , \_ )= person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | … |
| 24 | depositor(“Chalyy ”, BankT, \_ , \_ )=  =car("Chalyy", "hoenday solrais", "blue", 1000000000)    Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | … |
| 28 | depositor(“Chalyy ”, BankT, \_ , \_ )=  = depositor("Chalyy", "Sberbank", "812476672", 100210304)  Унификация успешна  Θ = {“Chalyy ” = Surname, “Moscow ” = City, StreetT= “Borisovo ”, PhoneT = "89261422462", BankT = “Sberbank”} | **Вывод:**  StreetT= “Borisovo ”, PhoneT = "89261422462", BankT = “Sberbank”  Откат, переход к следующему предложению. |
| 29 | depositor(“Chalyy ”, BankT, \_ , \_ )=  = depositor("Chalyy", "Finbank", "581888", 25)  Унификация успешна  Θ = {“Chalyy ” = Surname, “Moscow ” = City, StreetT= “Borisovo ”, PhoneT = "89261422462", BankT = “Finbank”} | **Вывод:**  StreetT= “Borisovo ”, PhoneT = "89261422462", BankT = “Finbank”  Откат, переход к следующему предложению. |
| 30 | depositor(“Chalyy ”, BankQ, \_ , \_ )=  = depositor("Kulikov", "Sberbank", "5233622", 120000)  Различаются константы типа surname  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | ... |
| 34 | depositor(“Chalyy ”, BankQ, \_ , \_ )= getCarOwnerByPhone(Phone, Surname, Brand, Price)  Разные функторы.  Унификация не успешна | Откат, переход к следующему предложению. |
| 35 | depositor(“Chalyy ”, BankQ, \_ , \_ )= getModelByPhone(Phone, Brand)  Разные функторы.  Унификация не успешна | Откат, переход к следующему предложению. |
| 36 | depositor(“Chalyy ”, BankQ, \_ , \_ )= getModelByPhone(Phone, Brand)  Разные функторы.  Унификация не успешна | **Новое состояние резольвенты:**  person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))  depositor(“Chalyy ”, BankT, \_ , \_ ) |
| 37 | person(“Chalyy ”, PhoneT, address(“Moscow ”, StreetT, \_ , \_ ))  =person("Chalyy", "8233551236", address("Moscow", "Borisovo", 25, 120))  Унификация успешна  Θ = {“Chalyy ” = Surname, “Moscow ” = City, StreetT= “Borisovo ”, PhoneT = "8233551236", BankT = Bank} | Новое состояние резольвенты:  depositor(“Chalyy ”, BankQ, \_ , \_ ) |
| 38 | depositor(“Chalyy ”, BankT, \_ , \_ )= person("Chalyy", "89261422462", address("Moscow", "Borisovo", 25, 120))  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | **…** |
| 43 | depositor(“Chalyy”, BankQ, \_ , \_ )=  = car("Chalyy", "hoenday solrais", "blue", 1000000000)    Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | **…** |
| 47 | depositor(“Chalyy”, BankQ, \_ , \_ )=  = depositor("Chalyy", "Sberbank", "812476672", 100210304)  Унификация успешна  Θ = {“Chalyy” = Surname, “Moscow ” = City, StreetT= “Borisovo ”, PhoneT = "8233551236", BankT = “Sberbank”} | **Вывод:**  StreetT= “Borisovo ”, PhoneT = "8233551236", BankT = “Sberbank”  Откат, переход к следующему предложению. |
| 48 | depositor(“Chalyy”, BankQ, \_ , \_ )=  = depositor("Chalyy", "Finbank", "581888", 25)  Унификация успешна  Θ = {“Chalyy” = Surname, “Moscow ” = City, StreetT= “Borisovo ”, PhoneT = "8233551236", BankT = “Finbank ”} | **Вывод:**  StreetQ= “Kovalenko”, PhoneQ = "89272030284", BankQ = “Finbank ”  Откат, переход к следующему предложению. |
| 49 | depositor(“Chalyy”, BankQ, \_ , \_ )=  = depositor("Kulikov", "Sberbank", "5233622", 120000)  Различаются константы типа surname  Унификация не успешна | Откат, переход к следующему предложению. |
| ... | ... | **...** |
| 53 | depositor(“Chalyy”, BankQ, \_ , \_ )= getCarOwnerByPhone(Phone, Surname, Brand, Price)  Разные функторы. Унификация не успешна | Откат, переход к следующему предложению. |
| 54 | depositor(“Chalyy”, BankQ, \_ , \_ )= getModelByPhone(Phone, Brand)  Разные функторы.  Унификация не успешна | Откат, переход к следующему предложению. |
| 55 | depositor(“Chalyy”, BankQ, \_ , \_ )= getInfoFromSurname(Surname, City, Street, Phone, Bank)  Разные функторы.  Унификация не успешна | **Новое состояние резольвенты:**  person(“Chalyy ”, Phone, address(“Borisovo ”, Street, \_ , \_ ))  depositor(“Chalyy ”, BankQ, \_ , \_ ) |
| 56 | person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))= person("Gag", "801239995", address("Mytischi", "boloto", 55, 33))  Разные константы типа surname  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | … |
| 58 | person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))=  car("Chalyy", "hoenday solrais", "blue", 1000000000)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | … |
| 62 | person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))=  depositor("Chalyy", "Sberbank", "812476672", 100210304)  Разные функторы  Унификация не успешна | Откат, переход к следующему предложению. |
| … | … | … |
| 68 | person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))= getCarOwnerByPhone(Phone, Surname, Brand, Price)  Разные функторы. Унификация не успешна | Откат, переход к следующему предложению. |
| 69 | person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))= getModelByPhone(Phone, Brand)  Разные функторы.  Унификация не успешна | Откат, переход к следующему предложению. |
| 70 | person(“Chalyy ”, Phone, address(“Moscow ”, Street, \_ , \_ ))= getInfoFromSurname(Surname, City, Street, Phone, Bank)  Разные функторы.  Унификация не успешна | **Новое состояние резольвенты:**  Резольвента пуста |

**Контрольные вопросы:**

1. **Что такое терм?**

Терм — это объект знаний в Prolog. Термы могут быть константами, переменными или составными термами. В свою очередь константы могут быть числовыми, символьными атомами или строками. Переменные могут быть именованными или анонимными. Составные термы используются для обозначения отношений между объектами, они объединяют отдельные элементы знаний в единый объект.

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

Предикат в матлогике — это функция, которая возвращает одно из двух значений: 0 («ложь») или 1 («Истина»)**.**

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

В Prolog предикат описывает отношение между объектами. Это отношение также как в математике может быть истинным.

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)?**

Виды предложений в программе:

* + Факты — с их помощью записываются знания.  
    Пример: car(petrov, volvo, 1223122, red).
  + Правила — это обобщенная запись знаний и условий, при которых знание является истиной.   
    Пример: getCarOwnerByPhone(Phone,Surname,Brand,Price):- person(Surname,Phone, \_ ),

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

* + Вопросы — используются для выяснения истинности какого-либо знания и

Пример: getCarOwnerByPhone("89261422462”, Surname, Brand, Price).

Основные предложения — это предложения, которые не содержат переменных.

Неосновные предложения — это предложения, которые содержат переменные в момент фиксации программы.

Синтаксически предложения записываются как

A :- B1,... , Bn.

А — это заголовок правила, а B1,..., Bn — тело правила.

Правила состоят и из заголовка и из тела. Факты — частный случай правил, состоят только из заголовка. Вопросы состоят только из тела.

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 предназначены для передачи знаний.

Переменные могут быть именованные и анонимные.

Если переменная не имеет значения, то она называется свободной, иначе — конкретизированной. Анонимные переменные не могут быть связаны со значением. Анонимные переменные используются в случаях, когда необходимо использовать переменную, однако ее значение не существенно.

Именованные переменные уникальны в рамках одного предложения, т. е. в разных предложениях одно и то же имя переменной может использоваться для обозначения разных объектов. Анонимные переменные уникальны везде.

Все переменные безтиповые, в процессе вычисления они могут связываться с любыми объектами.

Предложение содержащее переменные сформулировано в более общей-абстрактной форме, так как несколько знаний могут подойти под одно предложение.

1. **Что такое подстановка?**

Подстановкой называется множество пар, вида: { x i = t i } , где x i – переменная, а t i – терм. Каждой переменной xi в соответствие ставится терм ti.

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

Пример терма А — это терм B такой, что существует такая подстановка θ, что B = Aθ.

Примеры термов строятся в процессе унификации, когда происходит сравнение двух термов, с помощью подстановки всех возможных значений из базы знаний. Пример хранится в памяти для продолжения доказательства. Он уничтожается когда система заходит в тупик или на вопрос можно дать ответ «да».