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

к лабораторной работе №11-13

По курсу: «Функциональное и логическое программирование»

Темы: «Среда Visual Prolog5.2», «Структура программы на Prolog», «Работа программы на Prolog»

Студент: Якуба Д.В.
Группа: ИУ7-63Б
Преподаватели: Толпинская Н. Б.,
Строганов Ю. В.

Москва, 2021 г.

Практическая часть

Лабораторная работа 11.

Задание. Запустить среду Visual Prolog5.2. Настроить утилиту TestGoal. Запустить тестовую программу, проанализировать реакцию системы и множество ответов. Разработать свою программу – «Телефонный справочник». Протестировать работы программы.

```
predicates
    usesNumber(string, string).

clauses
    usesNumber("Pavel Perestoronin", "+79132249931").
    usesNumber("Ukunsun Vladimirovich", "+79996663322").
    usesNumber("No Name", "Unknown").
    usesNumber("Giopika Vladislavovich", "+71581588").
    usesNumber("Somebody once told him", "Unknown").
    usesNumber("Alis Sukocheva", Number) :-
      usesNumber("Pavel Perestoronin", Number).

goal
    usesNumber("Alis Sukocheva", Number); % Number=+79132249931
    usesNumber("Giopika Vladislavovich", Number); % Number=+71581588
    usesNumber(LastName, "+79996663322"); % LastName=Ukunsun Vladimirovich
    usesNumber(LastName, "+79132249931");
    % LastName=Pavel Perestoronin
    % LastName=Alis Sukocheva
    usesNumber(LastName, "777"); % No solution
    usesNumber("Giopika 000", _) % No
    .
```

Лабораторная работа 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)ю Описать порядок формирования вариантов ответа.

Исходную базу знаний сформировать с помощью только фактов.

*Исходную базу знаний сформировать, используя правила.

*Разработать свою базу знаний (содержание произвольно).

```
domains
    identificator = integer.
    firstName, lastName, university = string.
```

```
predicates
    refersToName(identificator, firstName, lastName).
    studyIn(identificator, university).
    fromUniversity(university, identificator, firstName, lastName).

clauses
    refersToName(1, "Sergey", "Kononenko").
    refersToName(2, "Pavel", "Perestoronin").
    refersToName(3, "Alexey", "Rabinovich").
    refersToName(4, "Magerram", "Zeynalov").
    refersToName(5, "Dmitry", "Yakuba").
    refersToName(6, "Pavel", "Nahimov").

    studyIn(1, "BMSTU").
    studyIn(2, "BMSTU").
    studyIn(3, "BMSTU").
    studyIn(4, "MIREA").
    studyIn(5, "BMSTU").
    studyIn(2, "MSU").
    studyIn(4, "MIT").
    studyIn(2, "MIT").
    studyIn(6, "ITMO").
    % studyIn(6, University) :- studyIn(2, University), studyIn(4, University).

    fromUniversity(University, Identificator, FirstName, LastName) :- studyIn(I
dentificator, University), refersToName(Identificator, FirstName, LastName).

goal
    % Get students' names
    refersToName(1, FirstName, LastName); % FirstName=Sergey, LastName=Kononenk
o
    refersToName(3, FirstName, LastName); % FirstName=Alexey, LastName=Rabinovi
ch
    refersToName(777, FirstName, LastName); % No solution

    % Get students' identificators
    refersToName(Id, "Pavel", LastName); % Id=2, LastName=Perestoronin; Id=6, L
astName=Nahimov
    refersToName(Id, "Magerram", "Zeynalov"); % Id = 4
    refersToName(Id, "Unknown", "Unknown"); % No solution

    % Get students from universities
    studyIn(Id, "BMSTU"); % Id=1 Id=2 Id=3 Id=5
    studyIn(4, University); % University=MIREA University=MIT
    studyIn(Id, "MSU"); % Id=2

    % Get students from university by rule
    fromUniversity("BMSTU", Id, FName, LName);
    % Id=1, FName=Sergey, LName=Kononenko
    % Id=2, FName=Pavel, LName=Perestoronin
    % Id=3, FName=Alexey, LName=Rabinovich
    % Id=5, FName=Dmitry, LName=Yakuba
    fromUniversity("MSU", Id, FName, LName);
    % Id=2, FName=Pavel, LName=Perestoronin
    fromUniversity("MIT", Id, FName, LName);
    % Id=4, FName=Magerram, LName=Zeynalov
    % Id=2, FName=Pavel, LName=Perestoronin
```

```
    % Get universities of student by rule
    fromUniversity(University, 3, FName, LName); % University=BMSTU, FName=Alex
ey, LName=Rabinovich
    fromUniversity(University, Id, "Pavel", LName);
        % University=BMSTU, Id=2, LName=Perestoronin
    % University=MSU, Id=2, LName=Perestoronin
    % University=MIT, Id=2, LName=Perestoronin
    % University=ITMO, Id=6, LName=Nahimov
    fromUniversity(University, Id, "Magerram", "Zeynalov")
    % Id=4, FName=Magerram, LName=Zeynalov
    % Id=2, FName=Pavel, LName=Perestoronin
    .
```

Порядок формирования вариантов ответа для refersToName(1, FirstName, LastName) (нахождение имени студента по его идентификатору):

Система в базе знаний ищет значения FirstName и LastName, при котором можно будет утвердительно ответить на вопрос «в составном терме refersToName: identificator == 1?».

Порядок формирования вариантов ответа для studyIn(Id, "BMSTU") (нахождение студентов, обучающихся в ВУЗе):

Система в базе знаний ищет значения Id, при котором можно будет утвердительно ответить на вопрос «в составном терме studyIn: university == “BMSTU”?».

Порядок формирования вариантов ответа для studyIn(4, University) (нахождение ВУЗов, в которых обучается студент):

Система в базе знаний ищет значения University, при котором можно будет утвердительно ответить на вопрос «в составном терме studyIn: identificatory == 4?».

Порядок формирования вариантов ответа для fromUniversity("BMSTU", Id, FName, LName) (нахождение имени и фамилии студентов, обучающихся в ВУЗе):

Система в базе знаний ищет значения Id, FirstName, LastName, при которых можно будет утвердительно ответить на вопрос «в составном терме refersToName: identificator == identificator из составного терма studyIn, в котором university == “BMSTU”?».

Лабораторная работа 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 др.,
- «Вкладчики банков»: Фамилия, Банк, счет, сумма, др.

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

Используя правила, обеспечить возможность поиска:

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

```
domains
    % Phonebook
    lastName, telephoneNum = string.
    city, street, houseNum = string.
    flatNum = integer.
    address = address(city, street, houseNum, flatNum).

    % cars
    carBrand, carColor = string.
    carPrice = real.

    % bank's depositors
    bankName = string.
    depositId = integer.
    depositSum = real.

predicates
    hasPhone(lastName, telephoneNum, address).
    hasCar(lastName, carBrand, carColor, carPrice).
    hasDeposit(lastName, city, bankName, depositId, depositSum). % For second ex
we have city

    hasCarByPhoneBook(telephoneNum, lastName, carBrand, carPrice).
    hasDepositByLastNameCity(lastName, city, street, bankName, telephoneNum).

clauses
    hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya",
"28/77", 333)).
```

```
    hasPhone("Yakuba",    "+79161586666",    address("Whitechapel",    "Bitsevky
Lesopark", "derevo 1", 1)).
    hasPhone("Yakuba", "+66666666666", address("Whitechapel", "Diggs Road", "7",
666)).
    hasPhone("Kovalev",   "+79993332211",   address("Moscow",   "Lusinovskaya",
"12/2", 12)).
    hasPhone("Kovalev", "+37773892047", address("Moscow", "Paveletsky Proezd",
"1", 326)).
    hasPhone("Kovalev",   "+11111111111",   address("Nijnii   Novgorod",   "Not
Lusinovskaya", "2/12", 88)).
    hasPhone("Kovalev",   "+22222222222",   address("Nijnii   Novgorod",   "Not
Paveletsky Proezd", "-1", 22)).


    hasCar("Perestoronin", "Daewoo", "Silver", 500000).
    hasCar("Yakuba", "Volkswagen", "Yellow-Silver", 900000).
    hasCar("Kovalev", "Volvo", "Dark Blue", 2200000).
    hasCar("Kovalev", "Mercedes", "Light crema", 3000000).
    hasCar("Kovalev", "KIA", "Pink", 2200000).

    hasDeposit("Perestoronin", "Moscow", "Switzerland Bank", 111111, 1000000).
    hasDeposit("Perestoronin", "Moscow", "Bokmal Bank", 000001, 999999.23).
    hasDeposit("Yakuba", "Whitechapel" ,"Sektor Gaza Bank", 666666, 300).
    hasDeposit("Kovalev", "Moscow", "Sberbank", 123456, 123456789).
    hasDeposit("Kovalev", "Nijnii Novgorod", "Not Sberbank", 654321, 987654321).

    hasCarByPhoneBook(Telephone_,    LName_,    CarBrand_,    CarPrice_)    :-
hasCar(LName_, CarBrand_, _, CarPrice_),
                                                        hasPhone(LName_,
Telephone_, _).

    hasDepositByLastNameCity(LastName_, City_, Street_, Bank_, Telephone_) :-
hasDeposit(LastName_, City_, Bank_, _, _),

hasPhone(LastName_, Telephone_, address(City_, Street_, _, _)).

goal
    % get car by phone book
    hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice);
        % LName=Kovalev, CarBrand=Volvo, CarPrice=2200000
        % LName=Kovalev, CarBrand=Mercedes, CarPrice=3000000
        % LName=Kovalev, CarBrand=KIA, CarPrice=2200000
    hasCarByPhoneBook("+66666666666", LName, CarBrand, CarPrice);
        % LName=Yakuba, CarBrand=Volkswagen, CarPrice=900000
    hasCarByPhoneBook("+79161586666", LName, CarBrand, CarPrice);
        % LName=Yakuba, CarBrand=Volkswagen, CarPrice=900000
    hasCarByPhoneBook("+79991112233", LName, CarBrand, CarPrice);
        % LName=Perestoronin, CarBrand=Daewoo, CarPrice=500000

    % get only car brand using last one
    hasCarByPhoneBook("+79993332211", _, CarBrand, _);
        % CarBrand=Volvo
        % CarBrand=Mercedes
        % CarBrand=KIA
    hasCarByPhoneBook("+66666666666", _, CarBrand, _);
        % CarBrand=Volkswagen
    hasCarByPhoneBook("+79161586666", _, CarBrand, _);
        % CarBrand=Volkswagen
```

```
    hasCarByPhoneBook("+79991112233", _, CarBrand, _);
        % CarBrand=Daewoo

    % get address, banks and telephone
    hasDepositByLastNameCity("Kovalev", "Moscow", Street, Bank, Telephone);
      % Street=Lusinovskaya, Bank=Sberbank, Telephone=+79993332211
      % Street=Paveletsky Proezd, Bank=Sberbank, Telephone=+37773892047

    hasDepositByLastNameCity("Kovalev", "Nijnii Novgorod", Street, Bank,
Telephone);
        % Street=Not Lusinovskaya, Bank=Not Sberbank, Telephone=+11111111111
        %     Street=Not     Paveletsky     Proezd,     Bank=Not     Sberbank,
Telephone=+22222222222

    hasDepositByLastNameCity("Perestoronin", "Moscow", Street, Bank, Telephone);
        % Street=Golubinskaya, Bank=Switzerland Bank, Telephone=+79991112233
        % Street=Golubinskaya, Bank=Bokmal Bank, Telephone=+79991112233

    hasDepositByLastNameCity("Yakuba", "Whitechapel", Street, Bank, Telephone)
        %     Street=Bitsevky     Lesopark,     Bank=Sektor     Gaza     Bank,
Telephone=+79161586666
        % Street=Diggs Road, Bank=Sektor Gaza Bank, Telephone=+66666666666
  .
```

Словесное описание порядка поиска ответа на вопрос для задания 1.А.

`hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice)`

№ шага	Сравниваемые термы; результат; подстановка, если есть	Дальнейшие действия: прямой ход или откат (к чему приводит?)
1	Сравнение: hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)). Унификация неуспешна (несовпадение функторов).	Откат, переход к следующему предложению
2-7	…	…
8	Сравнение: hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice) = hasCar("Perestoronin", "Daewoo", "Silver", 500000). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
9-12	…	…
13	Сравнение: hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice) = hasDeposit("Perestoronin", "Moscow", "Switzerland Bank", 111111, 1000000). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
14-17	…	…
18	Сравнение:	***Новое состояние резольвенты:***

	hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice) = hasCarByPhoneBook(Telephone_, LName_, CarBrand_, CarPrice_). Унификация успешна Подстановка: {Telephone_="+79993332211", LName_=LName, CarBrand_=CarBrand, CarPrice_=CarPrice}	hasPhone(LName, "+79993332211", _), hasCar(LName, CarBrand, _, CarPrice)
19	Сравнение: hasPhone(LName, "+79993332211", _) и hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)). Унификация неуспешна (несовпадение термов)	Откат, переход к следующему предложению
20-21	…	…
22	Сравнение: hasPhone(LName, "+79993332211", _) = hasPhone("Kovalev", "+79993332211", address("Moscow", "Lusinovskaya", "12/2", 12)). Унификация успешна Подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand_=CarBrand, CarPrice_=CarPrice}	***Новое состояние резольвенты:*** hasCar(LName, CarBrand, _, CarPrice)
23	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)). Не унифицируемы (несовпадение функторов)	Откат, переход к следующему предложению
24-29	…	…
30	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasCar("Perestoronin", "Daewoo", "Silver", 500000). Унификация неуспешна (несовпадение термов)	Откат, переход к следующему предложению
31	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasCar("Yakuba", "Volkswagen", "Yellow-Silver", 900000). Унификация неуспешна (несовпадение термов)	Откат, переход к следующему предложению
32	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasCar("Kovalev", "Volvo", "Dark Blue", 2200000). Унификация успешна Подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand="Volvo", CarPrice=2200000}	***Новое состояние резольвенты:*** **ПУСТА** **Вывод: LName=Kovalev, CarBrand=Volvo, CarPrice=2200000** Откат, следующее предложение, новая подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand_=CarBrand, CarPrice_=CarPrice}

33	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasCar("Kovalev", "Mercedes", "Light crema", 3000000). Унификация успешна Подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand="Mercedes", CarPrice=3000000}	***Новое состояние резольвенты:*** **ПУСТА** **Вывод: LName=Kovalev, CarBrand=Mercedes, CarPrice=3000000** Откат, следующее предложение, новая подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand_=CarBrand, CarPrice_=CarPrice}
34	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasCar("Kovalev", "KIA", "Pink", 2200000). Унификация успешна Подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand=" KIA ", CarPrice=2200000}	***Новое состояние резольвенты:*** **ПУСТА** **Вывод: LName=Kovalev, CarBrand=KIA, CarPrice=2200000** Откат, следующее предложение, новая подстановка: {Telephone="+79993332211", LName=”Kovalev”, CarBrand_=CarBrand, CarPrice_=CarPrice}
35	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasDeposit("Perestoronin", "Moscow", "Switzerland Bank", 111111, 1000000). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
36-39	…	…
40	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasCarByPhoneBook(Telephone_, LName_, CarBrand_, CarPrice_). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
41	Сравнение: hasCar(“Kovalev”, CarBrand, _, CarPrice) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone). Унификация неуспешна (несовпадение функторов)	Откат, достижение конца БЗ, переход к следующему предложению относительно шага 22 Новая подстановка: {Telephone_="+79993332211", LName_=LName, CarBrand_=CarBrand, CarPrice_=CarPrice}
42	Сравнение: hasPhone(LName, "+79993332211", _) = hasPhone("Kovalev", "+37773892047", address("Moscow",	Откат, переход к следующему предложению

	"Paveletsky Proezd", "1", 326)). Унификация неуспешна (несовпадение термов)	
43-44	…	…
45	Сравнение: hasPhone(LName, "+79993332211", _) = hasCar("Perestoronin", "Daewoo", "Silver", 500000). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
46-49	…	…
50	Сравнение: hasPhone(LName, "+79993332211", _) = hasDeposit("Perestoronin", "Moscow", "Switzerland Bank", 111111, 1000000). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
51-54	…	…
55	Сравнение: hasPhone(LName, "+79993332211", _) = hasCarByPhoneBook(Telephone_, LName_, CarBrand_, CarPrice_). Унификация неуспешна (несовпадение функторов)	Откат, переход к следующему предложению
56	Сравнение: hasPhone(LName, "+79993332211", _) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone). Унификация неуспешна (несовпадение функторов)	Откат, достижение конца БЗ, переход к следующему предложению относительно шага 18 Подстановки более нет
57	Сравнение: hasCarByPhoneBook("+79993332211", LName, CarBrand, CarPrice) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone). Унификация неуспешна (несовпадение функторов)	Достижение конца БЗ, резольвента пуста, завершение работы

Словесное описание порядка поиска ответа на вопрос для задания 1.В.

`hasCarByPhoneBook("+66666666666", _, CarBrand, _)`

№ шага	Сравниваемые термы; результат; подстановка, если есть	Дальнейшие действия: прямой ход или откат (к чему приводит?)
1	Сравнение: hasCarByPhoneBook("+66666666666", _, CarBrand, _) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)) не унифицируемы.	Откат, переход к следующему предложению
2-17	…	…
18	Сравнение: hasCarByPhoneBook("+66666666666", _, CarBrand, _) = hasCarByPhoneBook(Telephone, LName, CarBrand, CarPrice) унифицируемы. Связывание: Telephone = "+66666666666"	Прямой ход. унификация hasPhone(LName, "+66666666666", _)

19	Сравнение: hasPhone(LName, "+66666666666", _) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)) не унифицируемы.	Откат, переход к следующему предложению.
20	…	…
21	Сравнение: hasPhone(LName, "+66666666666", _) = hasPhone("Yakuba", "+66666666666", address("Whitechapel", "Diggs Road", "7", 666)) унифицируемы. Связывание: LName = "Yakuba"	Прямой ход, унификация hasCar(“Yakuba”, CarBrand, _, _)
22	Сравнение: hasCar(“Yakuba”, CarBrand, _, _) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)) не унифицируемы	Откат, переход к следующему предложению
23-28	…	…
29	Сравнение: hasCar(“Yakuba”, CarBrand, _, _) = hasCar("Perestoronin", "Daewoo", "Silver", 500000) не унифицируемы	Откат, переход к следующему предложению
30	Сравнение: hasCar(“Yakuba”, CarBrand, _, _) = hasCar("Yakuba", "Volkswagen", "Yellow-Silver", 900000) унифицируемы Связывается: CarBrand=”Volkswagen”	**Вывод: CarBrand=VOLKSWAGEN** Развязывается: CarBrand Откат, следующее предложение
31	Сравнение: hasCar(“Yakuba”, CarBrand, _, _) = hasCar("Kovalev", "Volvo", "Dark Blue", 2200000) не унифицируемы	Откат, переход к следующему предложению
32-33	…	…
34	Сравнение: hasCar(“Yakuba”, CarBrand, _, _) = hasDeposit("Perestoronin", "Moscow", "Switzerland Bank", 111111, 1000000) не унифицируемы	Откат, переход к следующему предложению
35-39	…	…
40	Сравнение: hasCar(“Yakuba”, CarBrand, _, _) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone) не унифицируемы	Откат, следующее предложение относительно шага 21 Развязывается: LName
41	Сравнение:	Откат, переход к следующему предложению

	hasPhone(LName, "+66666666666", _) = hasPhone("Kovalev", "+79993332211", address("Moscow", "Lusinovskaya", "12/2", 12)) не унифицируемы	
42-55	…	…
56	Сравнение: hasPhone(LName, "+66666666666", _) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone) не унифицируемы	Откат, следующее предложение относительно шага 18
57	Сравнение: hasCarByPhoneBook("+66666666666", _, CarBrand, _) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone) не унифицируемы	Откат, вывод полученных результатов Развязывается: Telephone

Словесное описание порядка поиска ответа на вопрос для задания 2.

`hasDepositByLastNameCity("Perestoronin", "Moscow", Street, Bank, Telephone)`

№ шага	Сравниваемые термы; результат; подстановка, если есть	Дальнейшие действия: прямой ход или откат (к чему приводит?)
1	Сравнение: hasDepositByLastNameCity("Perestoronin", "Moscow", Street, Bank, Telephone) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)) не унифицируемы	Откат, переход к следующему предложению
2-18	…	…
19	Сравнение: hasDepositByLastNameCity("Perestoronin", "Moscow", Street, Bank, Telephone) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone) Связываются: LastName="Perestoronin" = City="Moscow"	Прямой ход, унификация hasPhone(“Perestoronin”, Telephone, address(“Moscow”, Street, _, _))
20	Сравнение: hasPhone(“Perestoronin”, Telephone, address(“Moscow”, Street, _, _)) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)) унифицируемы Связываются: Telephone="+79991112233" = Street="Golubinskaya"	Прямой ход, унификация hasDeposit(“Perestoronin”, “Moscow”, Bank, _, _)
21	Сравнение: hasDeposit(“Perestoronin”, “Moscow”, Bank, _, _) = hasPhone("Perestoronin", "+79991112233", address("Moscow", "Golubinskaya", "28/77", 333)) не унифицируемы	Откат, переход к следующему предложению
22-32	…	…

33	Сравнение: hasDeposit(“Perestoronin”, “Moscow”, Bank, _, _) = hasDeposit("Perestoronin", "Moscow", "Switzerland Bank", 111111, 1000000) унифицируемы Связывается: Bank="Switzerland Bank"	**Вывод: Street=Golubinskaya, Bank=Switzerland Bank, Telephone=+79991112233** Развязывается: Bank Откат, следующее предложение
34	Сравнение: hasDeposit(“Perestoronin”, “Moscow”, Bank, _, _) = hasDeposit("Perestoronin", "Moscow", "Bokmal Bank", 000001, 999999.23) унифицируемы Связывается: Bank="Bokmal Bank"	**Вывод: Street=Golubinskaya, Bank=Bokmal Bank, Telephone=+79991112233** Развязывается: Bank Откат, следующее предложение
35	Сравнение: hasDeposit(“Perestoronin”, “Moscow”, Bank, _, _) = hasDeposit("Yakuba", "Whitechapel" ,"Sektor Gaza Bank", 666666, 300) не унифицируемы	Откат, переход к следующему предложению
36-39	…	…
40	Сравнение: hasDeposit(“Perestoronin”, “Moscow”, Bank, _, _) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone) не унифицируемы	Откат, следующее предложение относительно 20 Развязываются: Telephone, Street
41	Сравнение: hasPhone(“Perestoronin”, Telephone, address(“Moscow”, Street, _, _)) = hasPhone("Yakuba", "+79161586666", address("Whitechapel", "Bitsevky Lesopark", "derevo 1", 1)) не унифицируемы	Откат, переход к следующему предложению
42-57	…	…
58	Сравнение: hasPhone(“Perestoronin”, Telephone, address(“Moscow”, Street, _, _)) = hasDepositByLastNameCity(LastName, City, Street, Bank, Telephone) не унифицируемы	Откат, следующее предложение относительно 19
59	Сравнение: hasDepositByLastNameCity("Perestoronin", "Moscow", Street, Bank, Telephone) более не с чем сравнивать	Откат, вывод результатов Развязываются: LastName, City

Теоретическая часть

1. Что собой представляет программа на Prolog?

Ответ: программа на Prolog не является определением последовательности действий, она представляет собой набор фактов и правил, обеспечивающих получение заключений на основе этих утверждений.

Программа на Prolog представляет собой: базу знаний и вопрос. С помощью подбора ответов на запросы он извлекает хранящуюся информацию. База знаний содержит истинностные знания, используя которые программа выдает ответ на запрос. Одной из особенностей Prolog является то, что при поиске ответов на вопрос он рассматривает альтернативные варианты и находит все возможные решения (методом проб и ошибок) – множества значений переменных, при которых на поставленный вопрос можно ответить – «да».

2. Какова структура программы на Prolog?

Ответ: программа на Prolog состоит из разделов. Каждый раздел начинается со своего заголовка.

Структура программы:

- 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 не обязательно должны быть все разделы.

3. Как реализуется программа на Prolog? Как формируются результаты работы программы?

Ответ: Программа на Prolog представляет собой: базу знаний и вопрос. С помощью подбора ответов на запросы он извлекает хранящуюся информацию. База знаний содержит истинностные знания, используя которые программа выдает ответ на запрос. Одной из особенностей Prolog является то, что при поиске ответов на вопрос, он рассматривает альтернативные варианты и находит все возможные решения (методом проб и ошибок) – множества значений переменных, при которых на поставленный вопрос можно ответить – «да».

4. Что такое терм?

Ответ:

Терм – это:

1. Константа:
    - Число (целое, вещественное),
    - Символьный атом (комбинация символов латинского алфавита, цифр и символа подчеркивания, начинающихся со строчной буквы). Используется для обозначения конкретного объекта предметной области или для обозначения конкретного отношения,
    - Строка: последовательность символов, заключенных в кавычки,
2. Переменная:
    - Именованная – обозначается комбинацией символов латинского алфавита, цифр и символа подчеркивания, начинающейся с прописной буквы или символа подчеркивания,
    - Анонимная – обозначается символом подчеркивания,
3. Составной терм:
    - Это средство организации группы отдельных элементов знаний в единый объект, синтаксически представляется: **f(t1, t2, …, tm)**, где f – функтор (функциональный символ), t1, t2, …, tm – термы, в том числе и составные (их называют аргументами). Аргументом или параметром составного терма может быть константа, переменная или составной объект. Число аргументов предиката называется его арностью или местностью. Составные термы с одинаковыми функторами, но разной арности, обозначают разные отношения.

5. Что такое предикат в матлогике (математике)?

Ответ: предикат – это функция с множеством значений {0, 1} (или {ложь, истина}), определённая на множестве $M = M_1 \times M_2 \times ... \times M_n$. Таким образом, каждый набор элементов множества M характеризуется либо как «истинный», либо как «ложный».

6. Что описывает предикат в Prolog?

Ответ: предикаты – это утверждения программы. Фактически, структура предиката – это структура знания, отраженного в заголовке правил, входящих в процедуру.

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

Ответ:

Виды предложений: факты и правила.

Примеры из программы:

Факт: `hasCar("Perestoronin", "Daewoo", "Silver", 500000)`

Правило:
```
hasCarByPhoneBook(Telephone, LName, CarBrand, CarPrice) :-
hasCar(LName, CarBrand, _, CarPrice),
hasPhone(LName, Telephone, _)
```

Предложение называется основным в том случае, если оно не содержит переменных. В ином случае оно называется неосновным.

Синтаксис:

Правило: $A :- B_1, \ldots B_n.$,где A – заголовок правила, а $B_1, \ldots, B_n$ – тело правила.

Факт – это частный случай правила (без тела).

Заголовок, как составной терм, содержит знание о том, что между его аргументами существует отношение.

Таким образом основное предложение предназначено для описания отношений. Неосновные предложения предназначены для поиска ответа в базе данных.

8. 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?

Ответ: переменные предназначены для передачи информации «во времени и пространстве». Существуют анонимные и именованные переменные. В процессе выполнения программы именованные переменные могут связываться с различными объектами – конкретизироваться. Именованная переменная уникальна в пределах предложения. Анонимные переменные не могут быть связаны со значением. Любая анонимная переменная уникальна.

Предложение БЗ, содержащее переменные, является более общим, так как переменные не имеют значения и могут конкретизироваться в ходе работы системы.

9. Что такое подстановка?

Ответ: пусть дан терм

$$A(X_1, X_2, \ldots, X_n)$$

Подстановкой называется множество пар вида: $\{X_i = t_i\}$, где X_i – переменная, а t_i – терм.

10. Что такое пример терма? Как и когда строится? Как Вы думаете система строит и хранит термы?

Ответ:

Пусть $\omega = X_1 = t_1, X_2 = t_2, \dots, X_n = t_n$– подстановка. Тогда результат применения подстановки к терму обозначается: $A\omega$. Применение подстановки заключается в замене каждого вхождения переменной x_i на соответствующий терм. Терм B называется примером терма A, если существует такая подстановка ω, что $B = A\omega$.

В процессе выполнения программы система, используя встроенный алгоритм унификации, пытается обосновать возможность истинности вопроса, строя подстановки и примеры термов (вопроса и формулировки знания), используя базу знаний. Построение и подстановки производится путём конкретизации переменных. Сами термы хранятся в стеке.