

Министерство науки и образования РФ
Федеральное государственное автономное образовательное
учреждение высшего профессионального образования
«Санкт-Петербургский государственный электротехнический
университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)

Кафедра вычислительной техники

Отчёт
по лабораторной работе № 1
по дисциплине «Интеллектуальные системы»
Тема: «Разработка и отладка простой программы на языке Visual Prolog»

Выполнили студенты гр. 8307

Николаев Д.Е.
Репин С.А.
Такшеев А.Д.

Преподаватель

Родионов С.В.

Санкт-Петербург
2022г.

1. Результаты выполнения программы DOG.PRO.

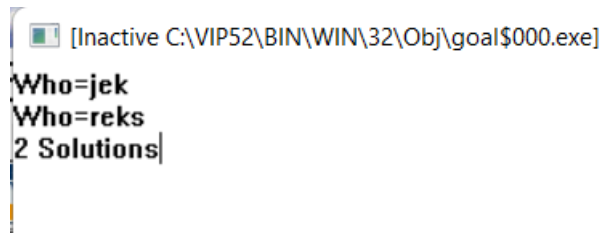


Рисунок 1.1 – результат выполнения программы DOG.PRO

Результатом программы стал вывод списка всех собак при выполнении запроса

dog(Who).

В секции предложений были записаны факты

dog("reks").
parent("jek", "reks").

Первый факт говорит о том, что reks – собака, а второй описывает родителя reks. Если посмотреть на правило вывода, то получится что jek тоже собака.

2. Трасса выполнения программы DOG.PRO с пояснениями.

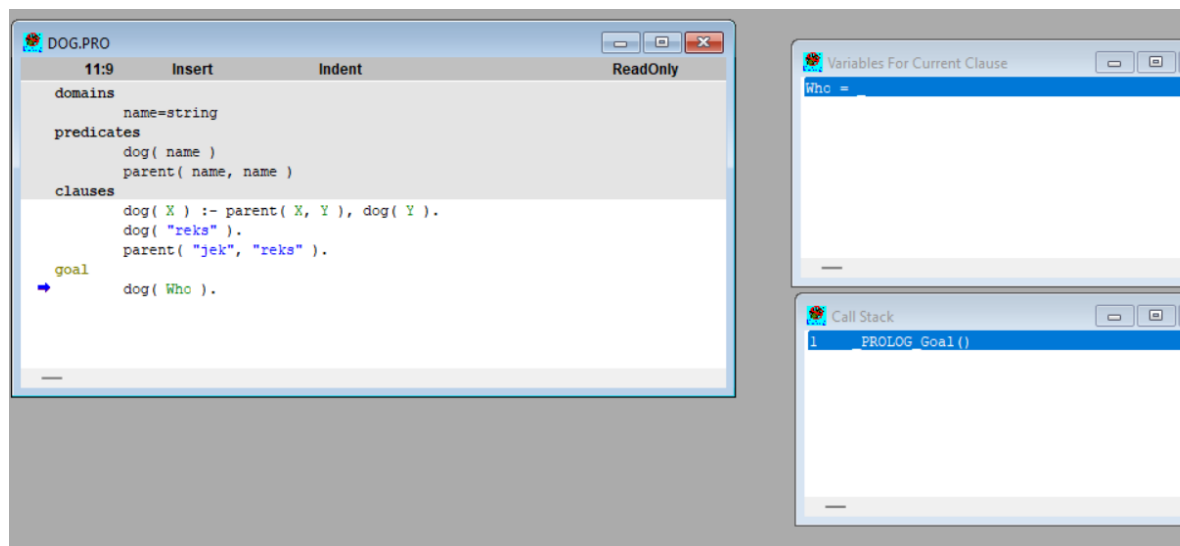


Рис. 2.1 – старт трассы

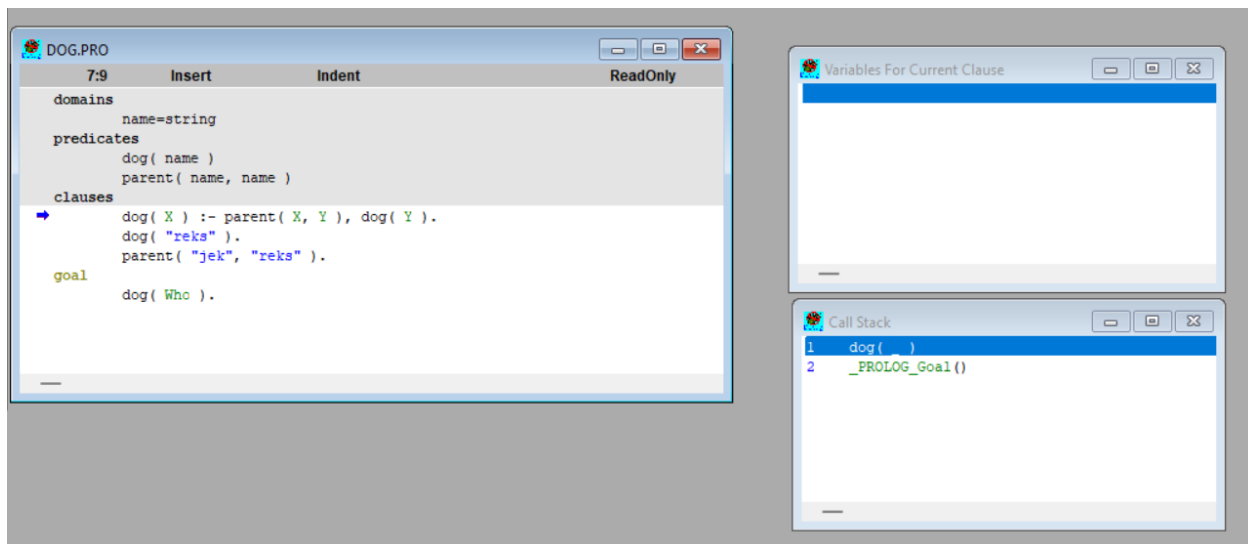


Рис.2.2 – вызываем любое правило/ факт с предикатом dog

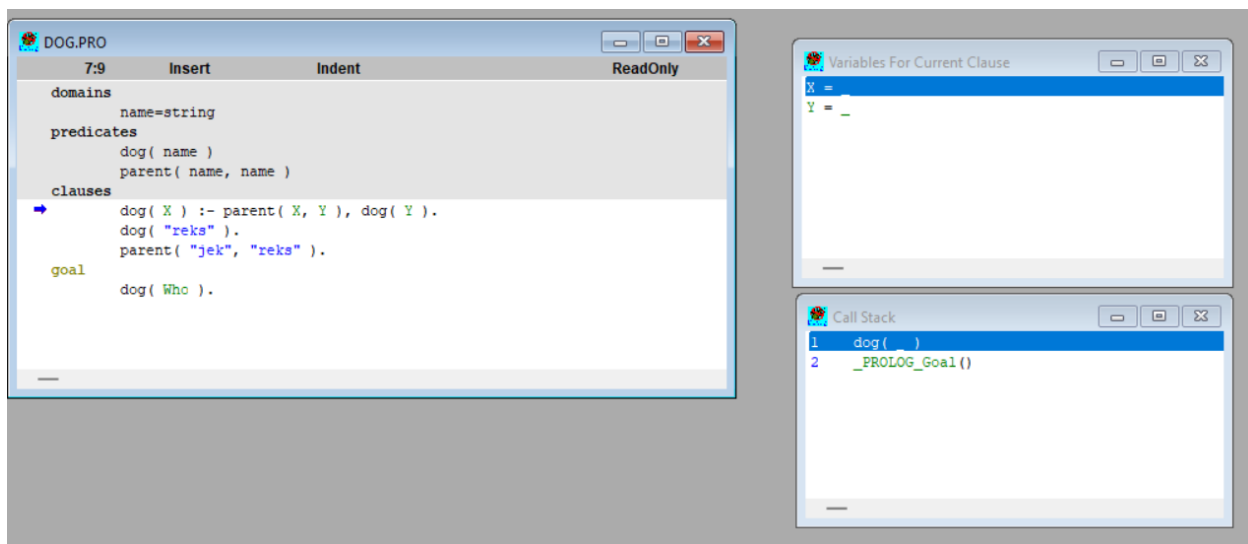


Рис. 2.3 – вызвали правило, чтобы проверить является ли X собакой, надо проверить является ли X родителем для собаки Y

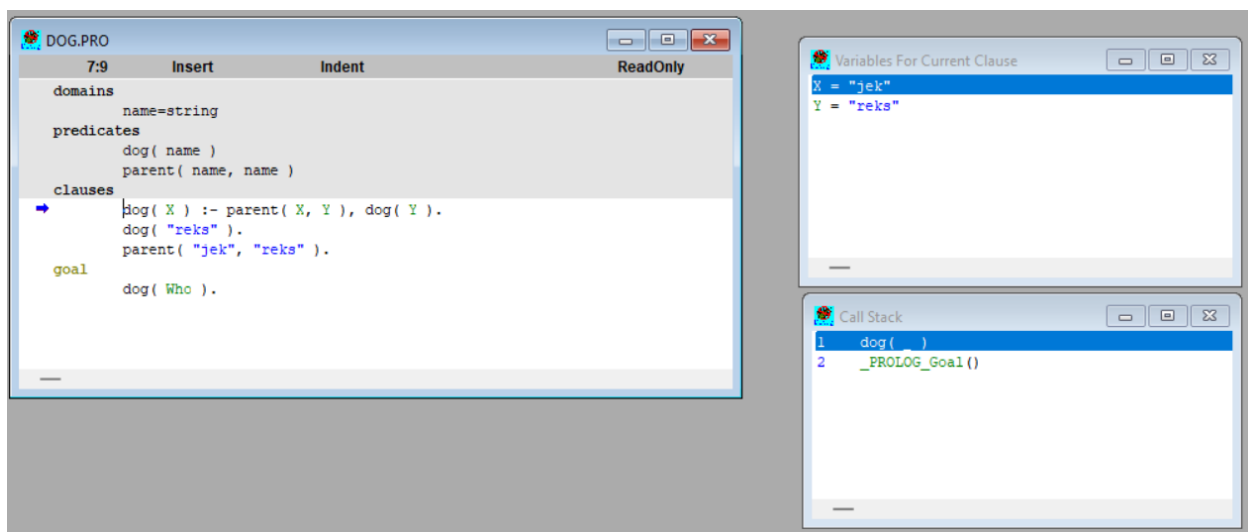


Рис. 2.4 – нашли факт о том что jek родитель reks

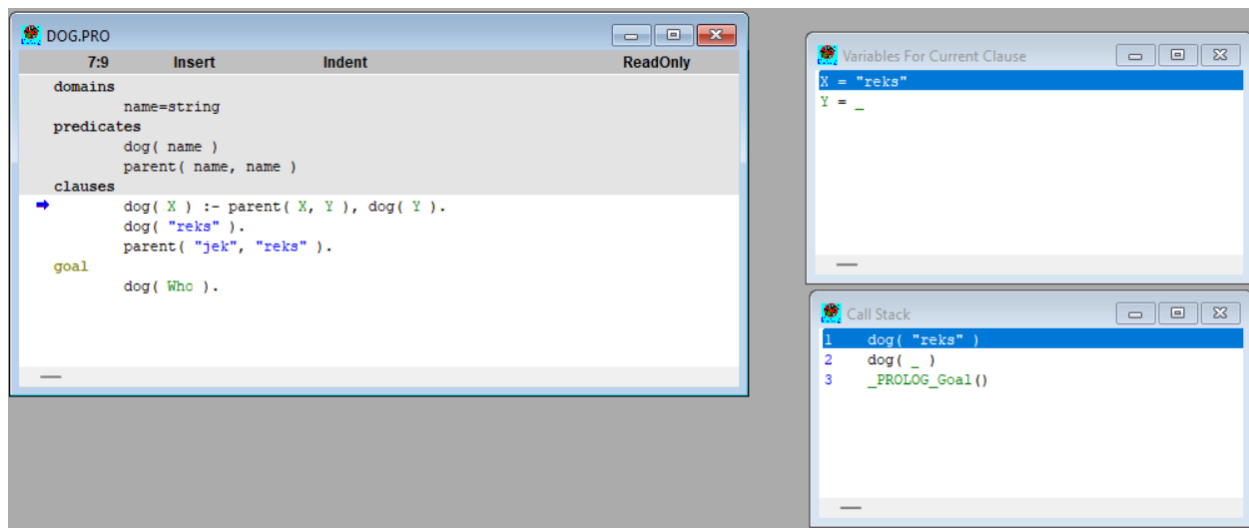


Рис 2.5 – теперь нужно проверить является ли reks родителем для какой-либо собаки

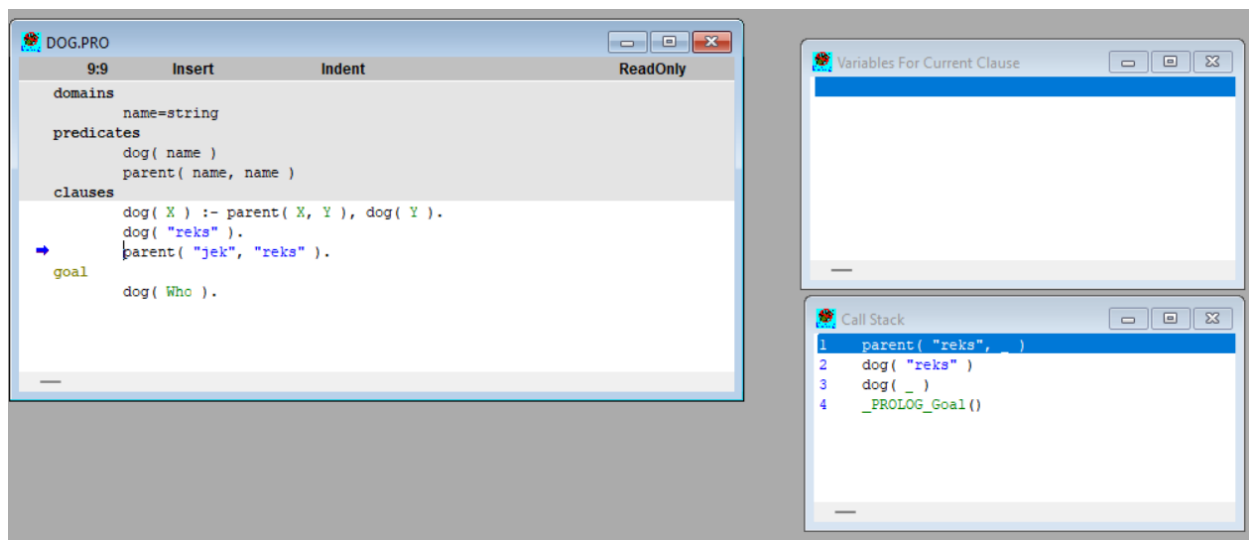


Рис 2.6 – reks родителем не является, но является собакой (есть факт об этом)

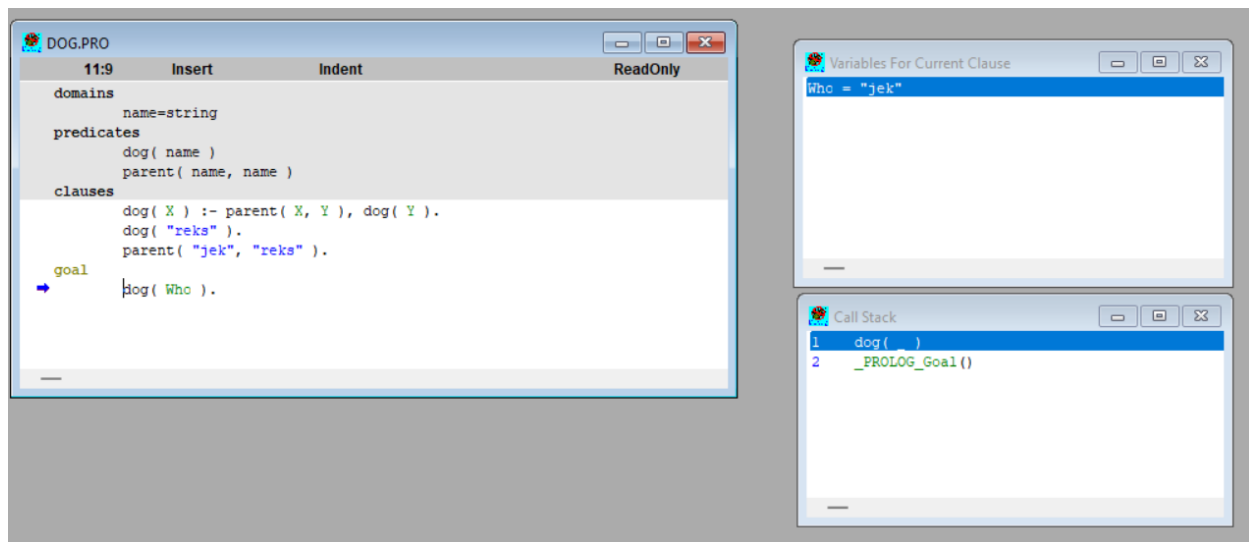
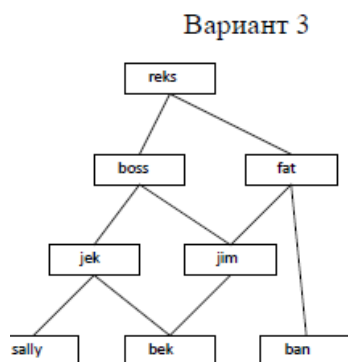


Рис. 2.7 – так как reks собака, а jek её родитель, то jek тоже является собакой

Больше нет фактов, которые могут активировать правила, поэтому программа завершается.

3. Текст программы DOG1.PRO, трассы выполнения запросов и объяснение результатов их выполнения.

Вариант 3



Текст программы:

```

% Номер варианта - 3
domains
    name=string
predicates
    dog( name )
    parent( name, name )
    grandparent( name, name )
clauses
    dog( X ) :-
        parent( X, Y ),
        dog( Y ).

    dog( "sally" ).
    dog( "bek" ).
    dog( "ban" ).
    dog( "fat" ).

    parent( "jek", "sally" ).
    parent( "jek", "bek" ).
    parent( "jim", "bek" ).
    parent( "boss", "jek" ).
  
```

```

parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

goal
% Кто является собакой ?
dog( who ).

% Кто является родителем ?
% parent( who, _ ).

% Кто является внуком (внучкой) ?
% grandparent( _, who ).

% Бек - собака ?
% dog( "bek" ).

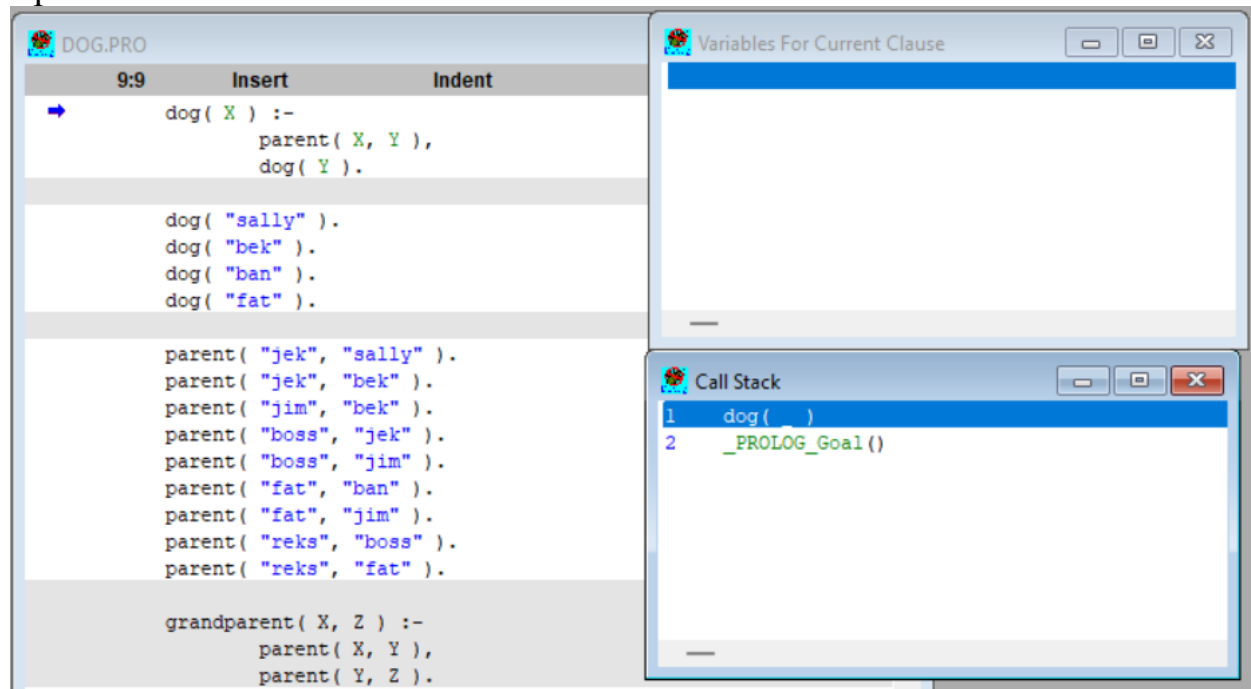
% Кто родитель собаки Бек ?
% parent( who, "bek" ).

% Кому Бек является родителем ?
% parent( "bek", who ).

```

Запрос Кто является собакой?

Трасса:



DOG.PRO

18:9 Insert Indent

```

dog( X ) :-
    parent( X, Y ),
    dog( Y ).

dog( "sally" ).
dog( "bek" ).
dog( "ban" ).
dog( "fat" ).

parent( "jek", "sally" ).
parent( "jek", "bek" ).
parent( "jim", "bek" ).
parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

```

Variables For Current Clause

Call Stack

1 parent(,)
2 dog(_)
3 _PROLOG_Goal()

DOG.PRO

11:17 Insert Indent ReadC

```

dog( X ) :-
    parent( X, Y ),
    dog( Y ).

dog( "sally" ).
dog( "bek" ).
dog( "ban" ).
dog( "fat" ).

parent( "jek", "sally" ).
parent( "jek", "bek" ).
parent( "jim", "bek" ).
parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

```

Variables For Current Clause

X = "jek"
Y = "sally"

Call Stack

1 dog(_)
2 _PROLOG_Goal()

DOG.PRO

13:9 Insert Indent ReadC

```

dog( X ) :-
    parent( X, Y ),
    dog( Y ).

dog( "sally" ).
dog( "bek" ).
dog( "ban" ).
dog( "fat" ).

parent( "jek", "sally" ).
parent( "jek", "bek" ).
parent( "jim", "bek" ).
parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

```

Variables For Current Clause

Call Stack

1 dog("sally")
2 dog(_)
3 _PROLOG_Goal()

Результат:

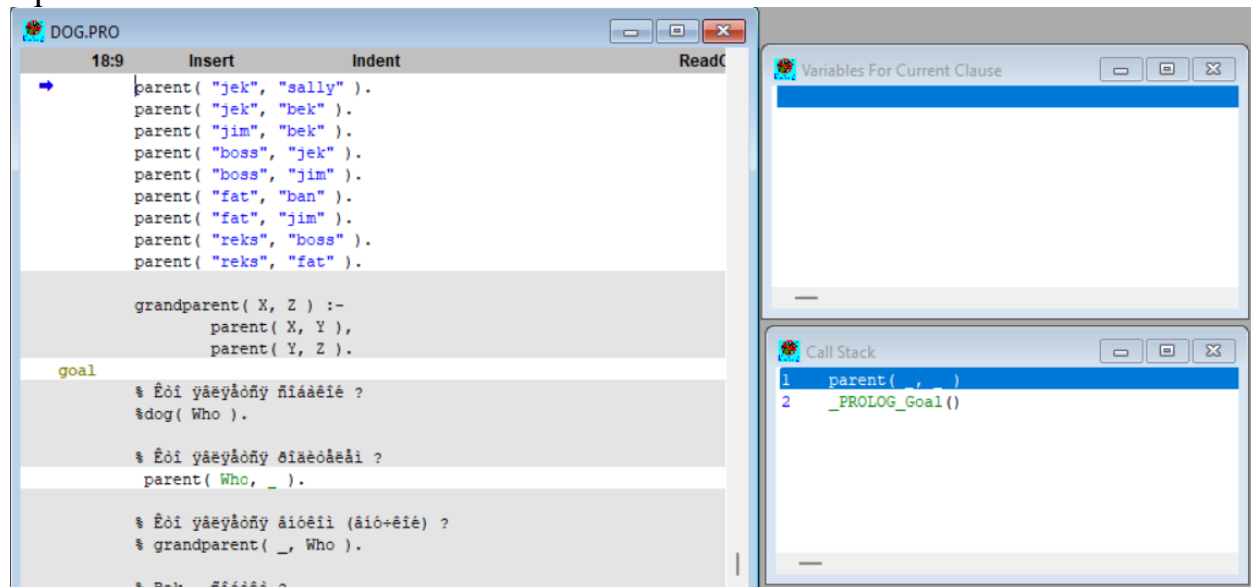
```

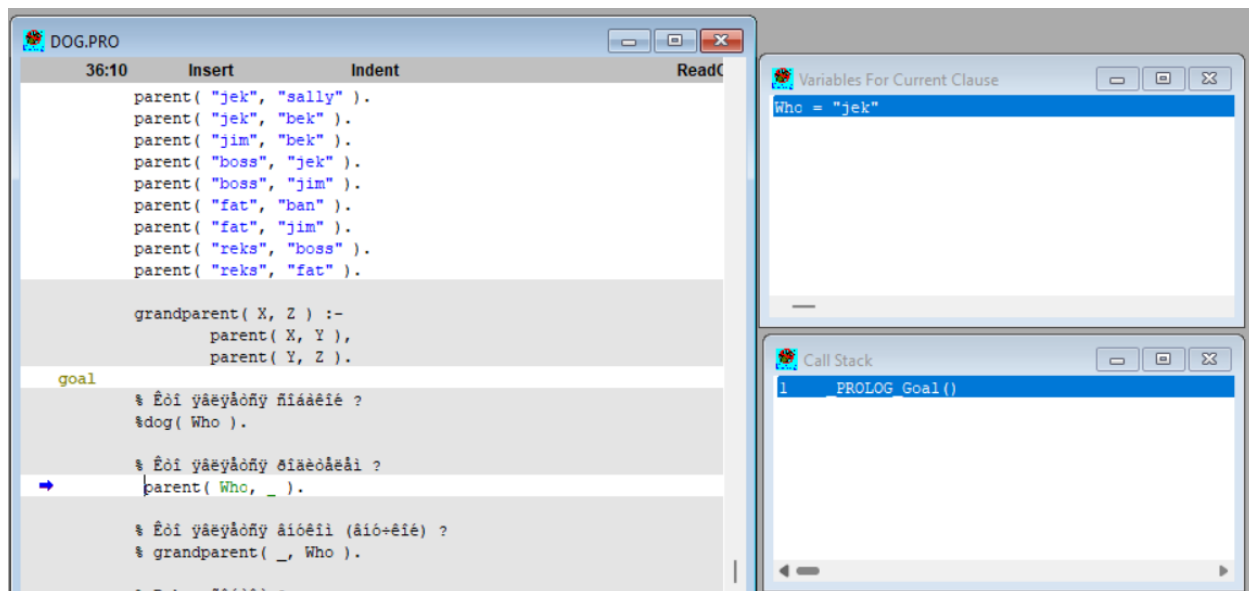
Who=jek
Who=jek
Who=jim
Who=boss
Who=boss
Who=boss
Who=fat
Who=fat
Who=reks
Who=reks
Who=reks
Who=reks
Who=reks
Who=reks
Who=sally
Who=bek
Who=ban
Who=fat
18 Solutions

```

Трасса аналогична выполнению программы DOG.PRO. Результатом стали 18 решений о том кто является собакой. Собаки повторяются так как в программе при нахождении собаки факт об этом не записывается, поэтому каждый раз нам необходимо узнавать кто является собакой родителем (самой старшей собакой).

Запрос Кто является родителем?
Трасса:





Результат:

```

[main@C:\V\1\32\BIN\WWW\32\OB\goal-prolog.exe]
Who=jek
Who=jek
Who=jim
Who=boss
Who=boss
Who=fat
Who=fat
Who=reks
Who=reks
9 Solutions
  
```

Результатом стал список всех родителей. Результатов 9, так как существует 9 фактов, удовлетворяющих запросу.

Запрос Кто является внуком (внучкой)?

Трасса:

DOG.PRO

39:10 Insert Indent ReadC

```

parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

goal
% Èòì yäëyâöñý ñíááëíé ?
% dog( Who ).

% Èòì yäëyâöñý ðíäëöäëäì ?
% parent( Who, _ ).

% Èòì yäëyâöñý äíóëíí (äíó+ëíé) ?
→ grandparent( _, Who ).

% Bek - ñíááëä ?
% dog( "bek" ).

% Èòì ðíäëöäëü ñíááëë Bek ?

```

Variables For Current Clause

Who = _

Call Stack

1 PROLOG_Goal()

DOG.PRO

29:17 Insert Indent ReadC

```

parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

→ grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

goal
% Èòì yäëyâöñý ñíááëíé ?
% dog( Who ).

% Èòì yäëyâöñý ðíäëöäëäì ?
% parent( Who, _ ).

% Èòì yäëyâöñý äíóëíí (äíó+ëíé) ?
grandparent( _, Who ).

% Bek - ñíááëä ?
% dog( "bek" ).

% Èòì ðíäëöäëü ñíááëë Bek ?

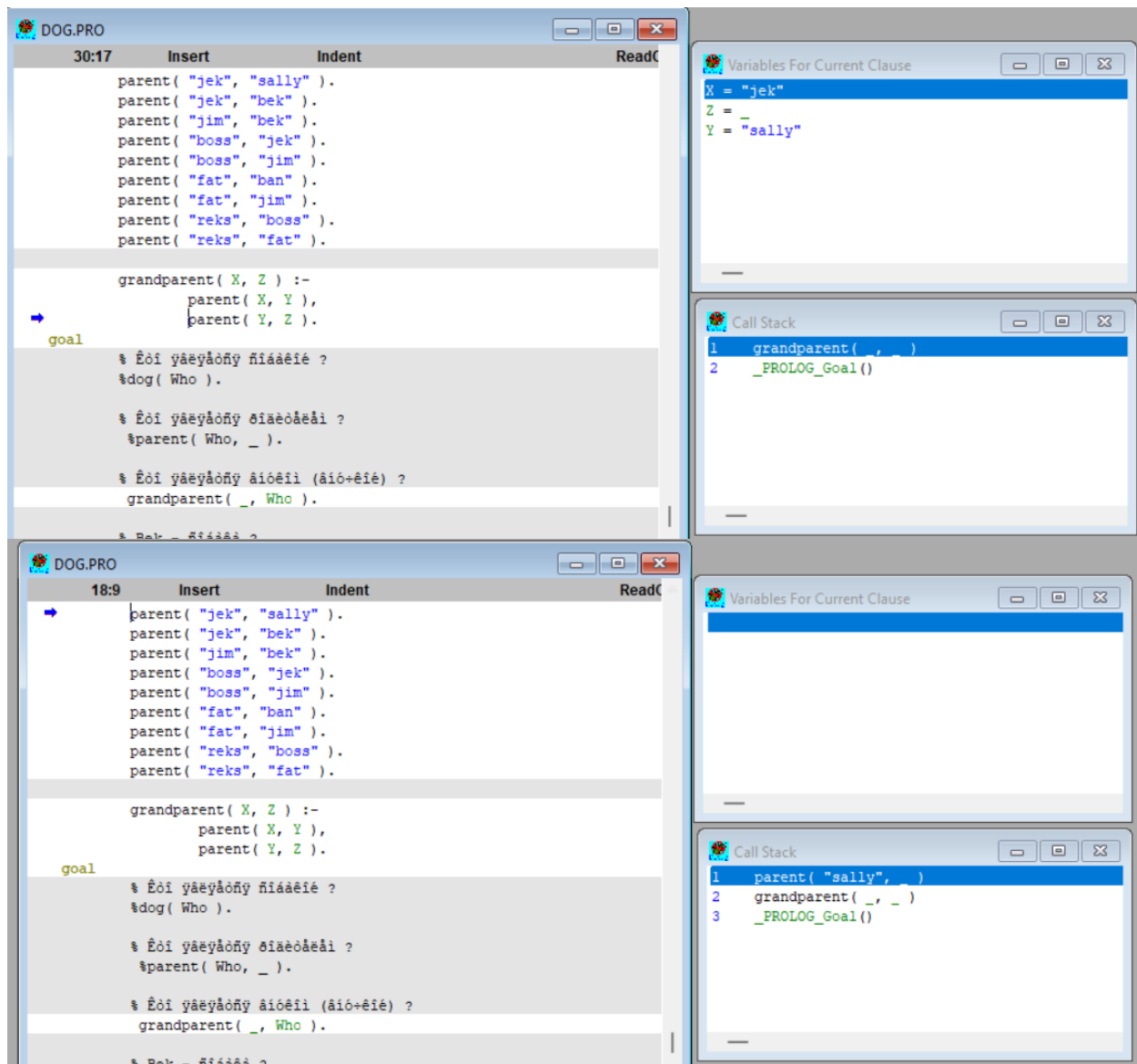
```

Variables For Current Clause

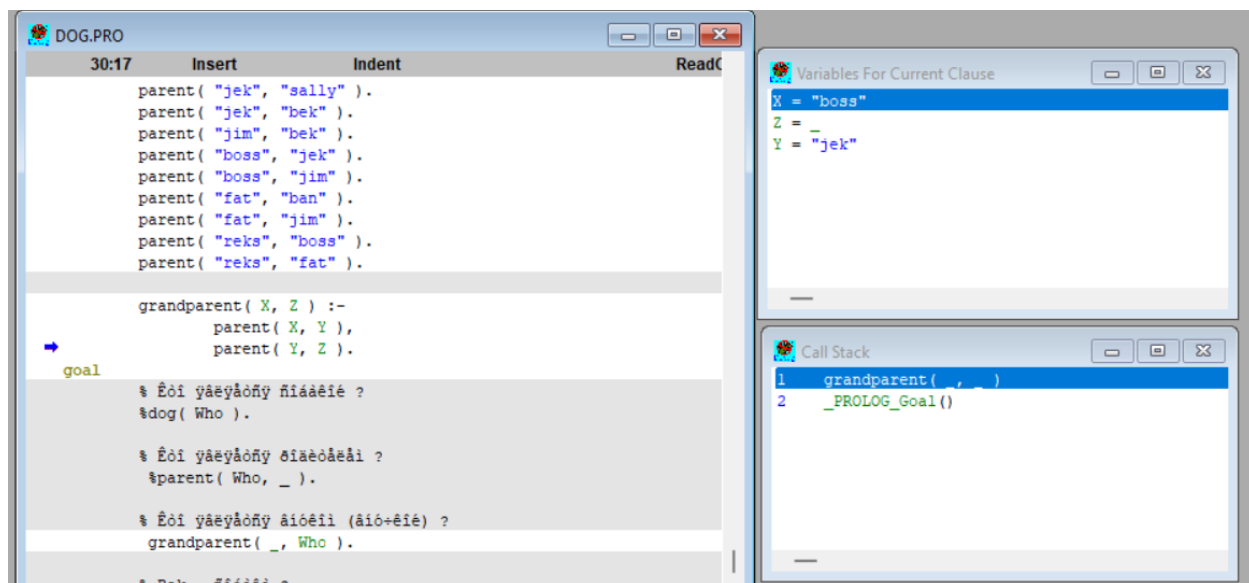
X = _
Z = _
Y = _

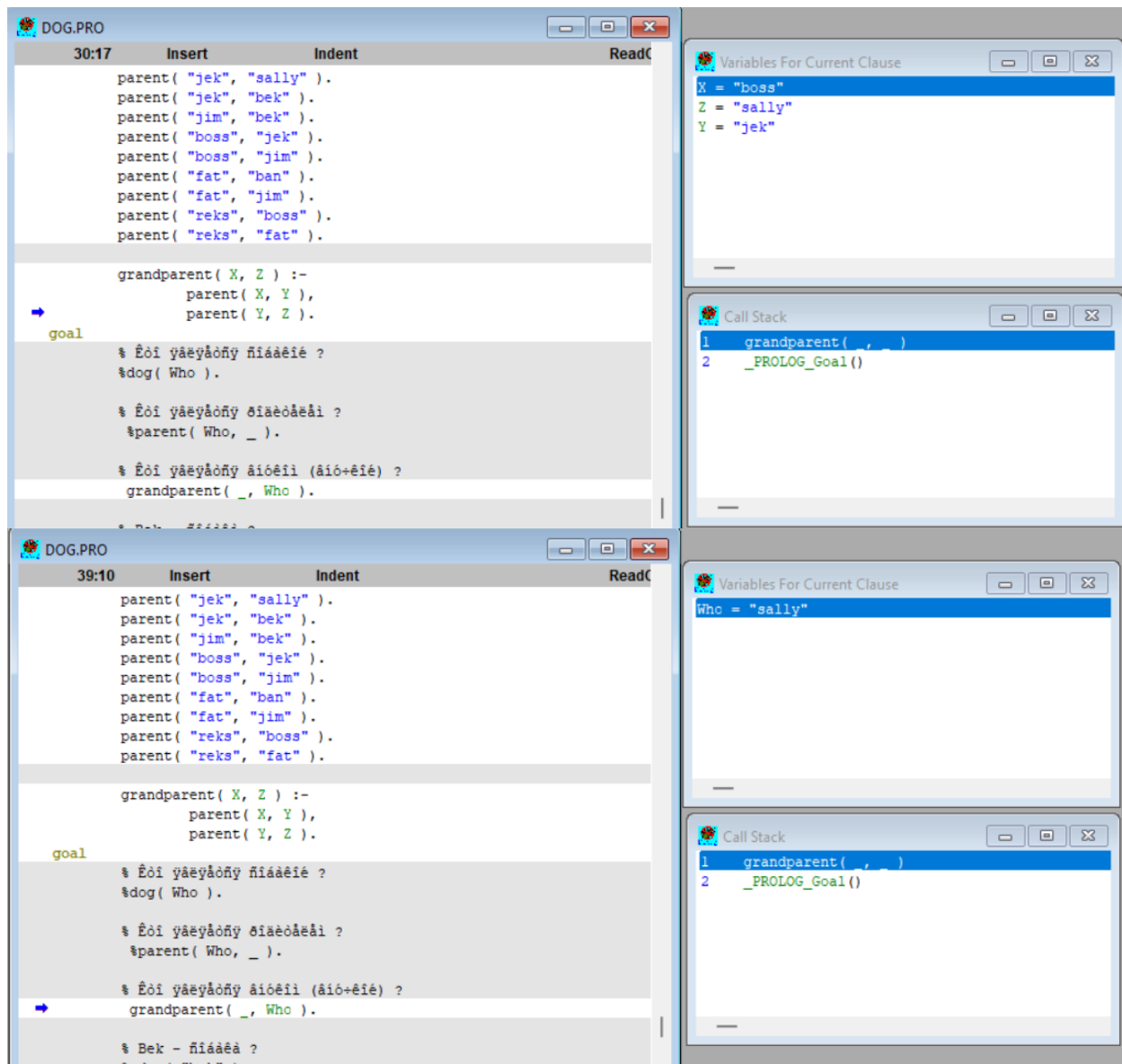
Call Stack

1 grandparent(_, _)
2 _PROLOG_Goal()



...





Результат:

```
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Who=sally
Who=bek
Who=bek
Who=bek
Who=jek
Who=jim
Who=ban
Who=jim
8 Solutions
```

Результатом стало 8 решений. Такое количество решений связано с количеством дедушек (бабушек), а точнее количеством связей между внуком (внучкой) и дедушкой (бабушкой). Так например bek записан 3 раза так как у него есть 3 связи с дедушкой (бабушкой): к boss через jek и jim, к fat через jim.

Запрос Век – собака?

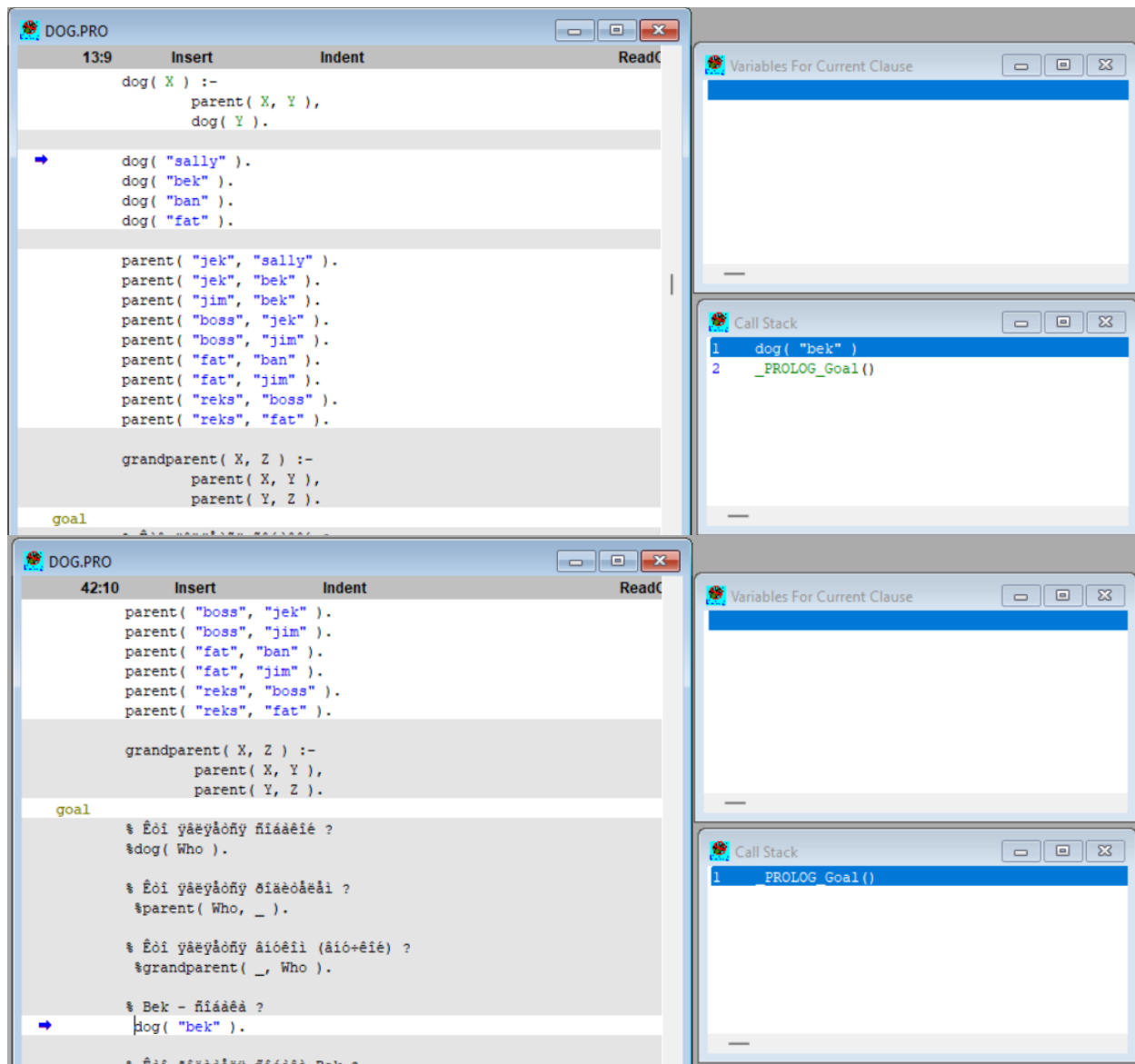
Трасса:

The image displays three sequential screenshots of a Prolog IDE window titled "DOG.PRO", illustrating the execution of a query. Each screenshot is accompanied by two auxiliary windows: "Variables For Current Clause" and "Call Stack".

First Screenshot (Top): The main window shows the initial state with a goal. The code includes a `parent` predicate, a `grandparent` predicate, and a `goal` clause. The goal is `% Bek - niaaëä ?` followed by `dog("bek").`. The auxiliary windows are empty.

Second Screenshot (Middle): The main window shows the first step of the execution. The goal is now `dog(X) :-` followed by `parent(X, Y),` and `dog(Y).`. The auxiliary windows show the current state: "Variables For Current Clause" has `X = "bek"` and `Y = _`; "Call Stack" has `1 dog("bek")` and `2 _PROLOG_Goal()`.

Third Screenshot (Bottom): The main window shows the final state after the query is completed. The goal is now `parent("jek", "sally").` followed by `parent("jek", "bek").` and `parent("jim", "bek").`. The auxiliary windows show the current state: "Variables For Current Clause" is empty; "Call Stack" has `1 parent("bek", _)`, `2 dog("bek")`, and `3 _PROLOG_Goal()`.



Результат:

[Inactive C:\]
yes

Результатом стал ответ yes – истина, то есть существует такой факт как «bek является собакой».

Запрос Кто родитель собаки bek?
Трасса:

DOG.PRO

45:10 Insert Indent ReadC

```

parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

goal
% Ёѡи яӕяѡѡы нѡӕӕѡѡ ?
%dog( Who ).

% Ёѡи яӕяѡѡы ѡѡӕѡӕӕѡ ?
%parent( Who, _ ).

% Ёѡи яӕяѡѡы ѡѡѡѡѡѡ (ѡѡ+ѡѡ) ?
%grandparent( _, Who ).

% Bek - нѡӕӕѡ ?
%dog( "bek" ).

% Ёѡи ѡѡӕѡӕѡ нѡӕӕѡ Bek ?
parent( Who, "bek" ).

```

Variables For Current Clause

Who =

Call Stack

1 PROLOG_Goal()

DOG.PRO

18:9 Insert Indent ReadC

```

parent( "jek", "sally" ).
parent( "jek", "bek" ).
parent( "jim", "bek" ).
parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

goal
% Ёѡи яӕяѡѡы нѡӕӕѡѡ ?
%dog( Who ).

% Ёѡи яӕяѡѡы ѡѡӕѡӕӕѡ ?
%parent( Who, _ ).

% Ёѡи яӕяѡѡы ѡѡѡѡѡѡ (ѡѡ+ѡѡ) ?
%grandparent( _, Who ).

```

Variables For Current Clause

Call Stack

1 parent(_, "bek")
2 _PROLOG_Goal()

DOG.PRO

45:10 Insert Indent ReadC

```

parent( "jek", "sally" ).
parent( "jek", "bek" ).
parent( "jim", "bek" ).
parent( "boss", "jek" ).
parent( "boss", "jim" ).
parent( "fat", "ban" ).
parent( "fat", "jim" ).
parent( "reks", "boss" ).
parent( "reks", "fat" ).

grandparent( X, Z ) :-
    parent( X, Y ),
    parent( Y, Z ).

goal
% Ёѡи яӕяѡѡы нѡӕӕѡѡ ?
%dog( Who ).

% Ёѡи яӕяѡѡы ѡѡӕѡӕӕѡ ?
%parent( Who, _ ).

% Ёѡи яӕяѡѡы ѡѡѡѡѡѡ (ѡѡ+ѡѡ) ?
%grandparent( _, Who ).

% Bek - нѡӕӕѡ ?

```

Variables For Current Clause

Who = "jek"

Call Stack

1 _PROLOG_Goal()

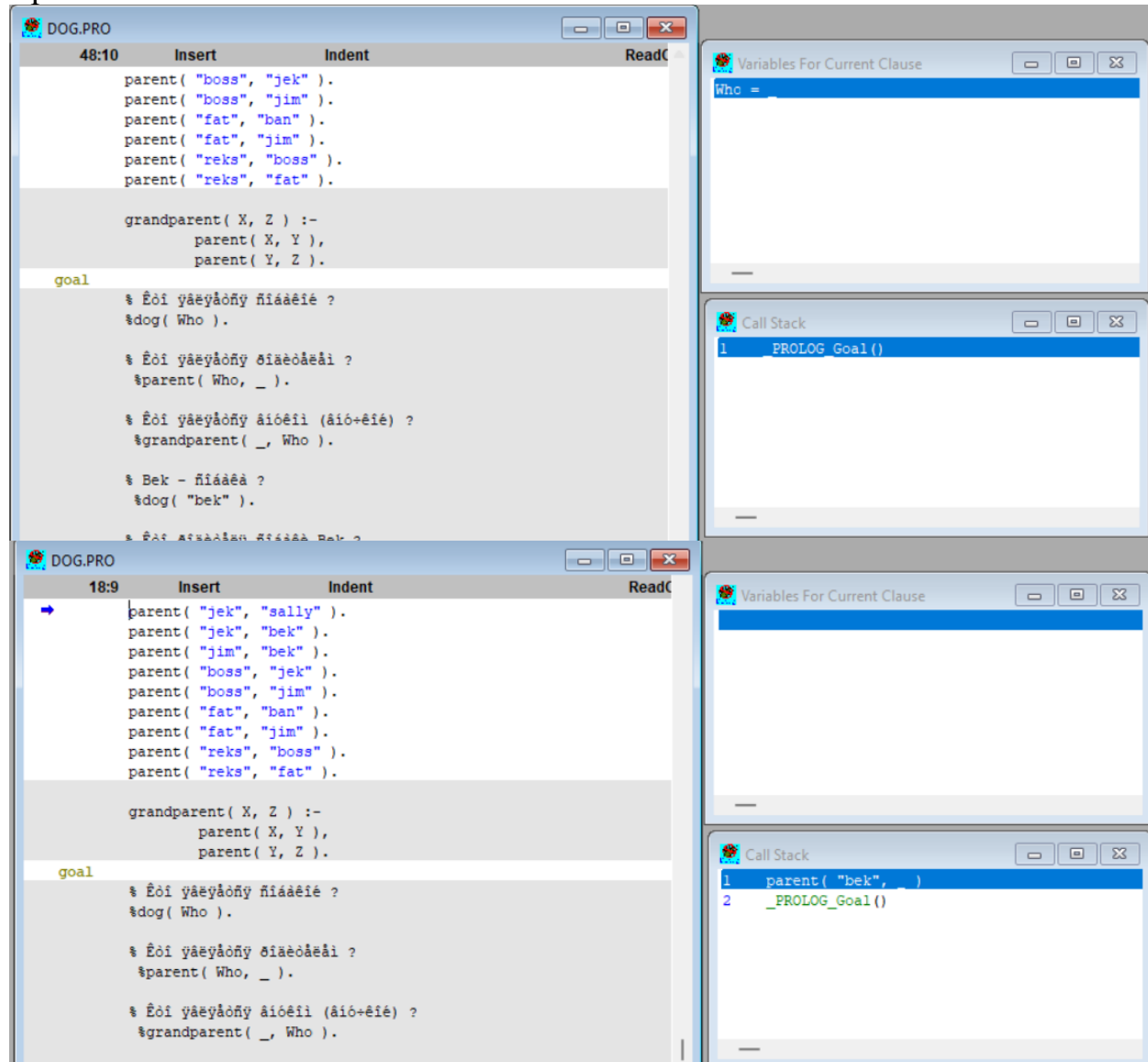
Результат:

```
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Who=jek
Who=jim
2 Solutions
```

Результатом стало два решения – родители bek

Запрос Кому Век является родителем?

Трасса:

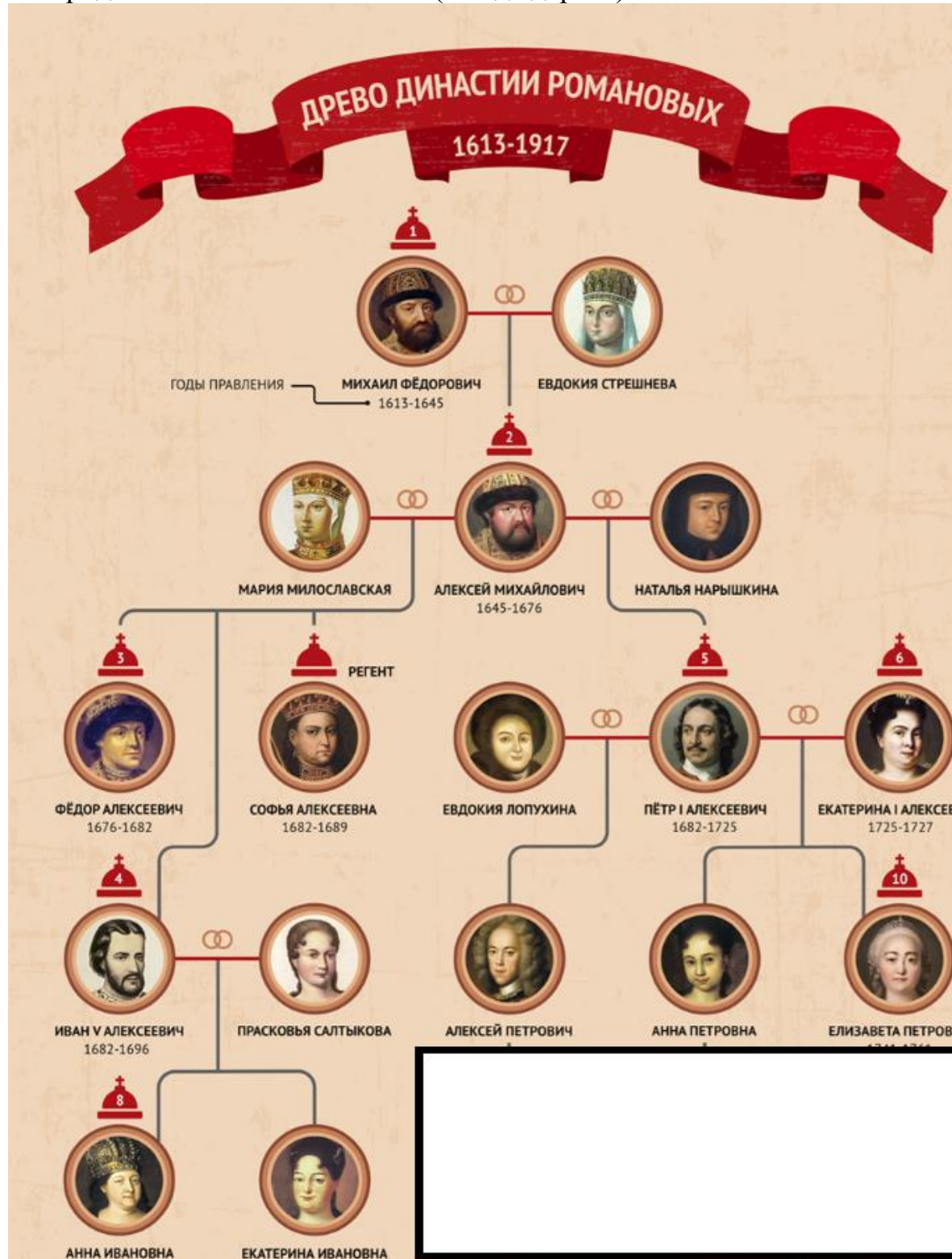


Результат:

```
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.
No Solution
```

Результатом стал ответ «Нет решения» так как у bek нет детей

4. Описание родственных связей в семье (в виде дерева).



5. Текст программы и результаты её работы для 5-6 разных запросов.

Текст программы:

domains

name=string

predicates

man(name)

woman(name)

parent(name, name)

```

married( name, name )

мать( name, name )
отец( name, name )
брат( name, name )
сестра( name, name )
дядя( name, name )
тетя( name, name )
племянник( name, name )
племянница( name, name )
дедушка( name, name )
бабушка( name, name )
внук( name, name )
внучка( name, name )
двоюродный_брат( name, name )
двоюродная_сестра( name, name )
clauses
man( "Михаил Федорович" ).
man( "Алексей Михайлович" ).
man( "Федор Алексеевич" ).
man( "Петр Алексеевич" ).
man( "Иван Алексеевич" ).
man( "Алексей Петрович" ).

woman( "Евдокия Стрешнева" ).
woman( "Мария Милославская" ).
woman( "Наталья Нарышкина" ).
woman( "Софья Алексеевна" ).
woman( "Евдокия Лопухина" ).
woman( "Екатерина Алексеевна" ).
woman( "Прасковья Салтыкова" ).
woman( "Анна Петровна" ).
woman( "Елизавета Петровна" ).
woman( "Анна Ивановна" ).
woman( "Екатерина Ивановна" ).

parent( "Михаил Федорович", "Алексей Михайлович" ).
parent( "Евдокия Стрешнева", "Алексей Михайлович" ).
parent( "Мария Милославская", "Федор Алексеевич" ).
parent( "Алексей Михайлович", "Федор Алексеевич" ).
parent( "Мария Милославская", "Софья Алексеевна" ).
parent( "Алексей Михайлович", "Софья Алексеевна" ).
parent( "Мария Милославская", "Иван Алексеевич" ).
parent( "Алексей Михайлович", "Иван Алексеевич" ).
parent( "Алексей Михайлович", "Петр Алексеевич" ).
parent( "Наталья Нарышкина", "Петр Алексеевич" ).
parent( "Петр Алексеевич", "Алексей Петрович" ).
parent( "Евдокия Лопухина", "Алексей Петрович" ).
parent( "Петр Алексеевич", "Анна Петровна" ).
parent( "Екатерина Алексеевна", "Анна Петровна" ).
parent( "Петр Алексеевич", "Елизавета Петровна" ).
parent( "Екатерина Алексеевна", "Елизавета Петровна" ).
parent( "Иван Алексеевич", "Анна Ивановна" ).
parent( "Прасковья Салтыкова", "Анна Ивановна" ).
parent( "Иван Алексеевич", "Екатерина Ивановна" ).
parent( "Прасковья Салтыкова", "Екатерина Ивановна" ).

married( "Михаил Федорович", "Евдокия Стрешнева" ).
married( "Алексей Михайлович", "Мария Милославская" ).
married( "Алексей Михайлович", "Наталья Нарышкина" ).
married( "Иван Алексеевич", "Прасковья Салтыкова" ).
married( "Петр Алексеевич", "Евдокия Лопухина" ).
married( "Петр Алексеевич", "Екатерина Алексеевна" ).

отец( X, Y ) :-
    parent( X, Y ),
    man( X ).

мать( X, Y ) :-
    parent( X, Y ),
    woman( X ).

```

```

брат( X, Y ) :-
    parent( Z, X ),
    parent( Z, Y ),
    man( X ),
    X <> Y.

сестра( X, Y ) :-
    parent( Z, X ),
    parent( Z, Y ),
    woman( X ),
    X <> Y.

дядя( X, Y ) :-
    брат( X, Z ),
    parent( Z, Y ).
дядя( X, Y ) :-
    married( X, Z ),
    тетя( Z, Y ).

тетя( X, Y ) :-
    сестра( X, Z ),
    parent( Z, Y ).
тетя( X, Y ) :-
    married( X, Z ),
    дядя( Z, Y ).

племянник( X, Y ) :- man( X ), дядя( Y, X ).
племянник( X, Y ) :- man( X ), тетя( Y, X ).

племянница( X, Y ) :- woman( X ), дядя( Y, X ).
племянница( X, Y ) :- woman( X ), тетя( Y, X ).

дедушка( X, Y ) :- man( X ), parent( X, Z ), parent( Z, Y ).
бабушка( X, Y ) :- woman( X ), parent( X, Z ), parent( Z, Y ).

внук( X, Y ) :- man( X ), дедушка( Y, X ).
внук( X, Y ) :- man( X ), бабушка( Y, X ).

внучка( X, Y ) :- woman( X ), дедушка( Y, X ).
внучка( X, Y ) :- woman( X ), бабушка( Y, X ).

). двоюродный_брат( X, Y ) :- man( X ), дядя( Z, Y ), parent( Z, X
). двоюродный_брат( X, Y ) :- man( X ), тетя( Z, Y ), parent( Z, X
). двоюродная_сестра( X, Y ) :- woman( X ), дядя( Z, Y ), parent( Z,
X ). двоюродная_сестра( X, Y ) :- woman( X ), тетя( Z, Y ), parent( Z,
X ).
goal
%кто является двоюродной сестрой и для кого?
двоюродная_сестра( who, whose ).
%Мужчины в семье
%man( who ).
% Кто является сестрой и для кого?
%сестра( who, whose ).
%кто является племянником и для кого?
%племянник( who, whose ).
%является ли Пётр Алексеевич внуком?
%внук( "Петр Алексеевич", _ ).
%женат ли Алексей Михайлович?
%married( "Алексей Михайлович", _ ).

```

Результаты работы:

```

[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Who=Анна Петровна, Whose=Анна Ивановна
Who=Анна Петровна, Whose=Екатерина Ивановна
Who=Елизавета Петровна, Whose=Анна Ивановна
Who=Елизавета Петровна, Whose=Екатерина Ивановна
Who=Анна Ивановна, Whose=Алексей Петрович
Who=Анна Ивановна, Whose=Анна Петровна
Who=Анна Ивановна, Whose=Елизавета Петровна
Who=Екатерина Ивановна, Whose=Алексей Петрович
Who=Екатерина Ивановна, Whose=Анна Петровна
Who=Екатерина Ивановна, Whose=Елизавета Петровна
10 Solutions|

```

Рис 5.1 – Результат выполнения запроса «Кто является двоюродной сестрой и для кого?»

```

[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Who=Михаил Федорович
Who=Алексей Михайлович
Who=Федор Алексеевич
Who=Петр Алексеевич
Who=Иван Алексеевич
Who=Алексей Петрович
6 Solutions|

```

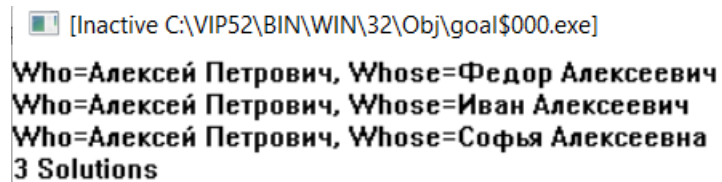
Рис 5.2 – Результат выполнения запроса «Мужчины в семье»

```

[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Who=Софья Алексеевна, Whose=Федор Алексеевич
Who=Софья Алексеевна, Whose=Иван Алексеевич
Who=Софья Алексеевна, Whose=Федор Алексеевич
Who=Софья Алексеевна, Whose=Иван Алексеевич
Who=Софья Алексеевна, Whose=Петр Алексеевич
Who=Анна Петровна, Whose=Алексей Петрович
Who=Анна Петровна, Whose=Елизавета Петровна
Who=Анна Петровна, Whose=Елизавета Петровна
Who=Елизавета Петровна, Whose=Алексей Петрович
Who=Елизавета Петровна, Whose=Анна Петровна
Who=Елизавета Петровна, Whose=Анна Петровна
Who=Анна Ивановна, Whose=Екатерина Ивановна
Who=Анна Ивановна, Whose=Екатерина Ивановна
Who=Екатерина Ивановна, Whose=Анна Ивановна
Who=Екатерина Ивановна, Whose=Анна Ивановна
15 Solutions|

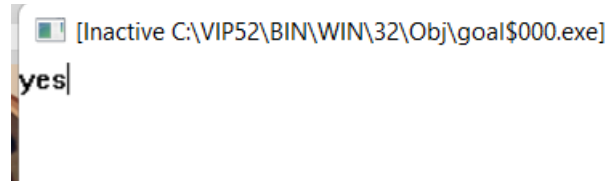
```

Рис 5.3 – Результат выполнения запроса «Кто является сестрой и для кого?»



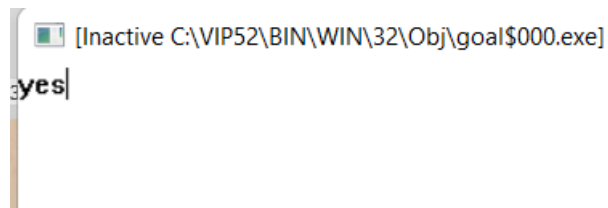
```
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
Who=Алексей Петрович, Whose=Федор Алексеевич
Who=Алексей Петрович, Whose=Иван Алексеевич
Who=Алексей Петрович, Whose=Софья Алексеевна
3 Solutions
```

Рис 5.4 – Результат выполнения запроса «Кто является племянником и для кого?»



```
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
yes|
```

Рис. 5.5 – Результат выполнения запроса «Является ли Петр Алексеевич внуком?»



```
[Inactive C:\VIP52\BIN\WIN\32\Obj\goal$000.exe]
yes|
```

Рис.5.6 – Результат выполнения запроса «Женат ли Алексей Михайлович?»