**The laboratory work 1 (Visual Prolog)**

Prolog is a language based on programming logic. The programmer in the language of Prolog focuses on describing the problem instead of detailed instructions prescribing how to solve this or that task. The Visual Prolog environment uses an approach called "visual programming", in which the appearance and behavior of programs are determined using special graphical design tools without traditional programming in the algorithmic language.

Visual Prolog automates the construction of complex procedures and frees the programmer from performing trivial operations. Using Visual Prolog, the design of the user interface and related windows, dialogs, menus, status notification lines, etc. is produced in a graphical environment. With the objects created, immediately the various Code Experts can work, which are used to generate basic and extended Prolog codes necessary to ensure their functioning.

Visual Prolog includes an interactive development environment (VDE), which includes text and different graphic editors, code generation tools that build control logic (Experts), and an extensible language interface for visual programming (VPI - Visual Programming Interface), Prolog-compiler, a set of various plug-in files and libraries, a link editor, files that contain examples and help.

Visual Prolog is supported by various operating systems, including MS-DOS PharLap-Extended DOS, all versions of Windows, 16- and 32-bit OS-2 target platforms, as well as some other systems that require a graphical user interface.

Depending on the chosen interface, the developer is provided access to a variety of code generators (Code Expert), various resource editors and special additional VPI predicates, definitions and libraries. Resource editors are used to create, link and edit windows, dialogs, menus, toolbars, help lines, string tables, shortcuts, cursors, bitmaps and online help. Code generators based on such structures create the necessary primary Prolog code. As a result, a primary code ("skeleton") appears, ready for compilation, editing bindings and execution.

The Visual Prolog interface includes: main menu, toolbar, project window. If there was an open project during the last use of the Visual Prolog system, the system will automatically reopen the project.

Figure 1 shows the Visual Prolog environment after startup. In the project window, the modules of the open project route.prj are displayed: karta.pro, route.pro, VPITools.pro

 

A window of a project

Buttons to work with project’s components

Project components

Code experts

рис.1. Среда разработки Visual Prolog 6.1

The left button bar in the project window allows you to select the required projectcomponent: module, window, menu, etc. Using the buttons of the right panel, the selected component can be edited (Edit button), deleted (Delete button), and added a new one (New button).
The File menu item contains commands for working with files. To create a new editing window, you can use the File | New. This command will create a new editor window with the title "NONAME".
The Edit menu contains commands that allow you to edit the text of the program. The built-in system editor on the interface is similar to the usual text editor. You can cut, copy and paste text, Cancel / Restore operations that you can activate from the Edit menu. Also, the Edit menu shows the "hot keys" associated for these actions.
The Project menu item contains commands for working with the project: create a new one, open it, start it, etc. The project can be launched by pressing the <R> button on the toolbar (or F9, or using the menu commands Project | Run).
The menu commands Options allow you to configure the project, set the necessary parameters.**Editing and testing programs in a Test mode**

The PROLOG program consists of sentences that can be facts, rules or queries. Typically, the program consists of four sections.
DOMAINS - section for describing domains (types). The section is used if non-standard domains are used in the program.
PREDICATES - a section for describing predicates. The section is used if non-standard predicates are used in the program.
CLAUSES - offers section. It is in this section that proposals are written: facts and rules of inference.
GOAL is the target section. In this section, a request is written.
The Visual Prolog environment allows you to test the program without creating a project. To do this, use the Test Goal utility. It is enough to create a new file, type the text of the program and activate Test Goal by pressing the <G> button on the toolbar. An autonomous executable file is not created. The Test Goal utility only compiles the code that is defined in the active editor window (the code in other open windows or project modules, if any, is ignored). Test Goal finds all possible solutions to the problem and automatically displays the values ​​of all variables.

**Example 1**

 There is a database that contains the following factors:

 Parent(“Ilya”, “Marina”)

 Parent(“Marina”, “Ira”)

 Parent(“Elena”, “Ivan”)

 Parent(“Nikolay”, “Ira”)

 Parent(“Olga”, “Alexei”)

 Parent(“Marina”, “Sasha”)

 Parent(“Sergei”, “Ivan”)

Identify:
1) Is it true that Marina is Sasha's parent?
2) Is it true that Alexei is Olga's parent?
3) who is the child of Nicholay;
4) who are the parents of Ivan;
5) all parents and their children.

**Decision**.
1. Start the Visual Prolog environment. Close the project window (if it is open) and open a new file (File | New) (Fig. 2)
In the window that appears, type the text of the program that contains sections: PREDICATES (description of the predicate parent), CLAUSES (lists available facts) and GOAL (query).

DOMAINS

 name=string

PREDICATES

 Nondeterm parent(name, name)

CLAUSES

 parent(“Ilya”, “Marina”).

 parent(“Marina”, “Ira”).

 parent(“Elena”, “Ivan”).

 parent(“Nikolay”, “Ira”).

 parent(“Olga”, “Alexei”).

 parent(“Marina”, “Sasha”).

 parent(“Sergei”, “Ivan”).

GOAL

 parent(“Marina”, “Sasha”).

Run and test the program using the Project | Test Goal (you can use the button on the toolbar <G> or the shortcut <Ctrl> + <G>). The result of the program will be displayed in a separate window

Note: before closing the program, close this window.
2. To answer the question: is it true that Alexei is the parent of Olga, change the query:

GOAL

parent(“Alexei”, “Olga”).

After launching the program (Project | Test Goal) you will get an answer:

no

3. To answer the question: who is the child of Nicholas, write down the goal:

GOAL

 Parent(“Nikolai”, X).

**The result:**

X=Ira

1 Solution

 4. To answer the question: who are the parents of Ivan, indicate the request:

GOAL

 Parent(X, “Ivan”), Parent(Y, “Ivan”), X<>Y.

**The result:**

X=Elena, Y=Sergei

X=Sergei, Y=Elena

2 Solutions

5. To identify all parents and their children, write down:

GOAL

Parent(X, Y).

**The result:**

X=Ilya, Y=Marina

X=Marina, Y=Ira

X=Elena, Y=Ivan

X=Nikolai, Y=Ira

X=Olga, Y=Alexei

X=Marina, Y=Sasha

X=Sergei, Y=Ivan

7 Solutions

Example 2
There are facts of the form: parent (name, name) and woman (name).
a) make up the rule of the mother and determine who is Masha's mother.

**The solution:**

DOMAINS

 name=string

PREDICATES

 parent(name, name)

 woman(name)

 mother(name,name)

CLAUSES

 parent("Marina","Ira").

 parent("Elena", "Anna").

 parent("Olga", "Marina").

 parent("Olga","Tatiyana").

 parent("Tatiyana","Katya").

 parent("Anna", "Masha").

 woman("Olga").

 woman ("Masha").

 woman ("Irina").

 woman ("Elena").

 woman ("Anna").

 woman ("Marina").

 woman ("Tatiyana").

 woman("Katya").

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

GOAL

 mother(X,"Masha").

**The result:**

X=Anna

1 Solution

Note: the keyword nondeterm defines non-deterministic predicates that can roll back and generate multiple solutions. Thus, if the problem involves the possibility of obtaining several solutions, it is necessary to declare the predicates as non-deterministic.

b) make up the granddaughter's rule and determine how many granddaughters Olga has and how her name is.

**Solution:**

DOMAINS

name=string

PREDICATES

nondeterm parent(name,name)

woman(name)

nondeterm mother(name,name)

nondeterm grandmother(name,name)

nondeterm granddaughter(name,name)

CLAUSES

 parent("Marina","Irina").

 parent ("Elena", "Anna").

 parent ("Olga", "Marina").

 parent ("Olga", "Tatiyana").

 parent ("Tatiyana", "Katya").

 parent ("Anna", "Masha").

 woman("Olga").

 woman("Masha").

 woman("Irina").

 woman("Elena").

 woman("Anna").

 woman("Marina").

 woman("Tatiyana").

 woman("Katya").

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

 grandmother(X,Z):-mother(X,Y),parent(Y,Z).

 granddaughter(X,Y):-grandmother(Y,X),woman(X).

GOAL

granddaughter(X, "Olga").

**Assignments:**

1. There is a database containing the following facts:

likes ("Anna," apples).
likes ("Sergey", bananas).
likes ("Andrew", apples).
likes ("Sveta", chocolate).
likes ("Vova", chocolate).
likes ("Anna", chocolate).
likes ("Vova", bananas).

loves ("Sveta", oranges).

Make a program that defines:
a) everyone who likes bananas;
b) who likes both chocolate and apples;
c) what Vova likes;
d) what both Sveta and Vova like.

2. There is a database containing the following facts:
plays ("Sasha", “football”).
plays ("Katya", “tennis”).
plays ("Sasha", “tennis”).
plays ("Andrew", “football”).
plays ("Oleg", “football”).
plays ("Olga", “tennis”).
plays ("Katya”, “volleyball”).
plays ("Oleg", “volleyball”).
Make a program that defines:
a) what kind of sport is Andrew interested in;
b) everyone who plays volleyball;
c) what kind of sport are Olga and Sasha interested in;
d) who is fond of football and volleyball.

3. There is a database containing the following facts:
likes ("Anne", “apples”).
likes ("Sergey", “bananas”).
likes ("Andrew", “apples”).
likes ("Light", “chocolate”).
likes ("Vova", “chocolate”).
likes ("Anna", “chocolate”).
loves ("Light", “oranges”).
likes ("Vova", “bananas”).
fruit (“apples”).
fruit (“bananas”).
fruit (“oranges”).
sweets (“chocolate”).
a) using the available facts, to compose a new rule for any fruits (X) and identify all those who like fruit;

b) using the available facts, create a new rule for any candy (X) and identify everyone who loves sweets;
c) using the available facts, make a rule of any taste (X) and identify everyone who likes fruits and sweets.

4. There is a database containing the following facts:
plays ("Sasha", “football”).
plays ("Katya", “tennis”).
plays ("Sasha", “tennis”).
plays ("Andrew", “football”).
plays ("Oleg", “football”).
plays ("Olga", “tennis”).
plays ("Katya", “volleyball”).
plays ("Oleg", “volleyball”).
woman ("Katya").
woman ("Olga").
the man ("Sasha").
the man ("Andrew").
the man ("Oleg").
a) using the available facts, create a new rule volleyball\_zhen (X) and identify all women playing volleyball;
b) using the available facts, draw up a new football-mug (X) rule and identify all men playing soccer;
c) Using the available facts, make a rule tennis\_para (X, Y), which allows you to find a mixed tennis couple (male + female). Identify all such pairs.