

Structure of the program in Prolog. Execution management





Peculiarities of Visual Prolog

- ❖ Visual Prolog is a compiled language
- ❖ Other versions have elements of interpretation of a code during the execution
- ❖ Strict data typing
- ❖ Rules are not data: you can't add or remove them during the execution
- ❖ You can't define new operations



Program sections

- ❖ compiler directives
- ❖ **CONSTANTS** – section of constants description
- ❖ **DOMAINS** – section of domains description
- ❖ **DATABASE** – section of description of internal database predicates
- ❖ **PREDICATES** – section of predicates description
- ❖ **CLAUSES** – section of clauses description
- ❖ **GOAL** – section of internal goal description



Peculiarities of Visual Prolog

- ❖ All sections can be in any order
- ❖ Predicates and domain must be defined before their usage
- ❖ Predicates declared in a section DATABASE can be added and removed from internal database while the execution



Program containing only the goal

GOAL

```
write("hello"), readchar(_).
```

Compiler directives



- ❖ trace – to trace the program
- ❖ nowarnings – to suppress the message that a variable occurs only once
- ❖ include – insertion of some file content
- ❖ check_determ – compulsory check of predicates determination



Peculiarities of Visual Prolog

- ❖ You can start tracing only for the definite predicate
- ❖ If there is a tracing the optimization of recursion is not working
- ❖ To establish non-determination of predicates by default: Options – Project – Compiler options – Warnings – Default predicate type – Nondeterm

Constants

CONSTANTS

$\pi=3.14$

`path="c:\\prolog\\bgi"`

Domains



- ❖ integer – an integer number (-32768...32767)
- ❖ real – a float number ($\pm e^{-307} \dots \pm e^{308}$)
- ❖ char – a symbol in apostrophes
- ❖ string – a sequence of symbols in double quotation marks
- ❖ symbol – a symbolic constant (atom)
- ❖ file – a file



Description of your own domain

$\langle \text{name_domain} \rangle = \langle \text{description_domain} \rangle$

or

file = $\langle \text{symbolic file name}_1 \rangle$; ...;
 $\langle \text{symbolic file name}_N \rangle$

or

$\langle \text{name_list_domain} \rangle = \langle \text{name_domain_of_list_elements} \rangle^*$

Examples:

DOMAINS

i=integer

list=i*



Description of a structured domain

$\langle \text{name_structure} \rangle = \langle \text{name_functor} \rangle$
 $(\langle \text{name_domain_first_component} \rangle, \dots,$
 $\langle \text{name_domain_last_component} \rangle) [;\langle \text{name_functor} \rangle(\dots)]^*$

Examples:

$\text{flatpoint} = \text{p}(\text{integer}, \text{integer})$

$\text{triangle} = \text{tr}(\text{point}, \text{point}, \text{point})$

$\text{fullpoint} = \text{p}(\text{integer}, \text{integer}); \text{p}(\text{integer}, \text{integer}, \text{integer})$

Description of predicates



`<name_predicate> (<name_first_argument>, ..., <name_last_argument>).`

Examples:

PREDICATES

`mother(string,string).`

`member(integer,integer*).`

`member(real,real*).`

`member(char,char*).`

`member(string,string*).`

One and the same
predicate will work with
different data of different
domains

Standard predicates



- ❖ `readln(_)`
- ❖ `readint(_)`
- ❖ `readreal(_)`
- ❖ `readchar(_)`
- ❖ `readterm(name_domain, term_domain)`
- ❖ `write([<variable / constant / value>, ...])`
- ❖ `writeln` – format output
- ❖ `nl`
- ❖ `upper_lower(_, _)`
- ❖ `str_int(_, _)`
- ❖ `str_real(_, _)`
- ❖ `str_char(_, _)`
- ❖ `char_int(_, _)`
- ❖ `true`
- ❖ `fail`
- ❖ `free(_)`
- ❖ `bound(_)`



Standard predicates

- ❖ `div()`
- ❖ `mod()`
- ❖ `trunc()`
- ❖ `round()`
- ❖ `random(_)`
- ❖ `random(<число>, _)`

- ❖ All embedded predicates are determined

Program “Relatives”



DOMAINS

s=string

PREDICATES

nondeterm mother(s,s)

nondeterm grandmother(s,s)

CLAUSES

mother("Наташа","Даша").

mother("Даша","Маша").

grandmother(X,Y):-

 mother(X,Z),

 mother(Z,Y).

Well-formed program:

- 1) Input an empty line between procedures
- 2) Start a body of a rule with an indent
- 3) Start each sub-goal from a new line with an indent

Print all grandmothers

Print all mothers

Print all mothers by pressing the button



Execution management

1. Method of a depth search (backtracking)
2. Method of a rollback after a failure
3. Cut and rollback
4. Method of a user-defined search



Backtracking

DOMAINS

s=string

PREDICATES

mother(s,s)

grandmother(s,s)

CLAUSES

mother("Dasha", "Masha").

mother("Natasha", "Dasha").

mother("Natasha", "Glasha").

mother("Dasha", "Sasha").

grandmother(X,Y):-

 mother(X,Z),

 mother(Z,Y).

GOAL

grandmother(B,V).



Method of a rollback after a failure

DOMAINS

s=string

PREDICATES

mother(s,s)

grandmother(s,s)

CLAUSES

mother("Dasha", "Masha").

mother("Natasha", "Dasha").

mother("Natasha", "Glasha").

mother("Dasha", "Sasha").

grandmother(X, Y):-

 mother(X, Z),

 mother(Z, Y).

GOAL

grandmother(B, V),

write("Grandmother's name – ", B),

write("Granddaughter's name – ", V).



Method of a rollback after a failure

DOMAINS

s=string

PREDICATES

mother(s,s)

grandmother(s,s)

CLAUSES

mother("Dasha", "Masha").

mother("Natasha", "Dasha").

mother("Natasha", "Glasha").

mother("Dasha", "Sasha").

grandmother(X, Y):-

 mother(X, Z),

 mother(Z, Y).

GOAL

grandmother(B, V),

write("Grandmother's name – ", B),

write("Granddaughter's name – ", V),
nl, fail.



Method of a rollback after a failure

DOMAINS

s=string

PREDICATES

mother(s,s)

grandmother(s,s)

show_names

CLAUSES

mother("Dasha","Masha").

mother("Natasha","Dasha").

mother("Natasha","Glasha").

mother("Dasha","Sasha").

grandmother(X,Y):-

mother(X,Z),

mother(Z,Y).

show_names:-

mother(_,Name),

write(" ", Name), nl,

fail.

GOAL

write("Daughters' names: "),nl,

show_names.



Method of a rollback after a failure

DOMAINS

s=string

PREDICATES

mother(s,s)

grandmother(s,s)

show_names2(s)

CLAUSES

mother("Dasha","Masha").

mother("Natasha","Dasha").

mother("Natasha","Glasha").

mother("Dasha","Sasha").

grandmother(X,Y):-

mother(X,Z),

mother(Z,Y).

```
show_names2(Mother):-  
    mother(M,Name),  
    M=Mother,  
    write(" ", Name), nl,  
    fail.
```

GOAL

```
write("Dasha's daughters names:  
"),nl,  
show_names2("Dasha").
```



Cut and rollback

DOMAINS

s=string

PREDICATES

mother(s,s)

grandmother(s,s)

show_names3(s)

CLAUSES

mother("Dasha","Masha").

mother("Natasha","Dasha").

mother("Natasha","Glasha").

mother("Dasha","Sasha").

grandmother(X,Y):-

 mother(X,Z),

 mother(Z,Y).

```
show_names3(Daughter):-  
    mother(_,Name),  
    write(" ", Name), nl,  
    Name=Daughter,  
    write("We found her!"),!.
```

GOAL

```
write("All daughters up to Glasha: "),nl,  
show_names3("Glasha").
```



Method of a user-defined search

repeat.

repeat:-

repeat.

double_char:-

repeat,

readchar(C),

write(C,C), nl,

C='.',!,

nl,write("You entered a full stop. You finished.").

GOAL

write("Enter symbols which you want to repeat. Full stop is to finish"),

nl,

double_char.