

## Lógica para computação - Prolog

Nome: Cristiano Koxne

Ra: 1920251

### 1) Pesquisar sobre Regras, Bases e Fatos em PROLOG.

O Prolog é uma linguagem declarativa que usa um fragmento da lógica de primeira ordem para representar o conhecimento sobre um dado problema. Um programa em Prolog é um conjunto de axiomas e de regras de inferência que descrevem um dado problema. A este conjunto chama-se normalmente base de conhecimento.

A execução de um programa em Prolog consiste na dedução de consequências lógicas da base de conhecimento. Prolog pesquisa a base de conhecimento à procura de axiomas e regras que permitam (por dedução lógica) dar uma resposta. O motor de inferência faz a dedução aplicando o algoritmo de resolução de primeira ordem.

#### Regras e comandos fundamentais

Números:

Operadores Aritméticos	
adição	+
subtração	-
multiplicação	*
divisão	/
divisão inteira	//
resto da divisão inteira	mod
potência	**
atribuição	is

Dados os números X e Y:

Operadores Relacionais	
X é maior que Y	X>Y
X é menor que Y	X<Y
X é maior e igual a Y	X>=Y
X é menor e igual a Y	X<=Y
X é igual a Y	X:=Y
X unifica com Y	X = Y
X é diferente de Y	X/=Y

#### Constantes

São cadeias compostas pelos seguintes caracteres:

- letras maiúsculas: A, B, ..., Z
- letras minúsculas: a, b, ..., z
- dígitos: 0, 1, ..., 9
- caracteres especiais: + - \* / < > = : . & \_ ~

**Variáveis (tipo de dado que pode receber algum valor)**

São cadeias de letras, dígitos e caracteres ' , sempre começando com letra maiúscula ou com o caractere ' \_ '.

Exemplos de variáveis:

X, Resultado, Objeto3, Lista\_Alunos, ListaCompras, \_x25, \_32.

## **Predicados e estruturas**

Podemos definir um tipo de dado a uma variável genérica, desta forma definindo uma árvore de relações entre variáveis.

Exemplo:

numero(X) .     *desta forma X passa a ser um número*

numero(Y).     *Y também passa a ser considerado número*

Podemos ainda definir uma estrutura inferindo vários valores a uma variável

data(D,M,2003) .     *nesse caso, atribuímos 3 variáveis ao predicado da data.*  
ponto(X,Y,Z).

Além disso podemos atribuir predicados de pertinência em variáveis conjuntas

Exemplo:

pai(jonas, marcos).

pai(jonas, ana).     *nesse caso, Jonas se tornou pai tanto de Ana como de*

*Marcos*

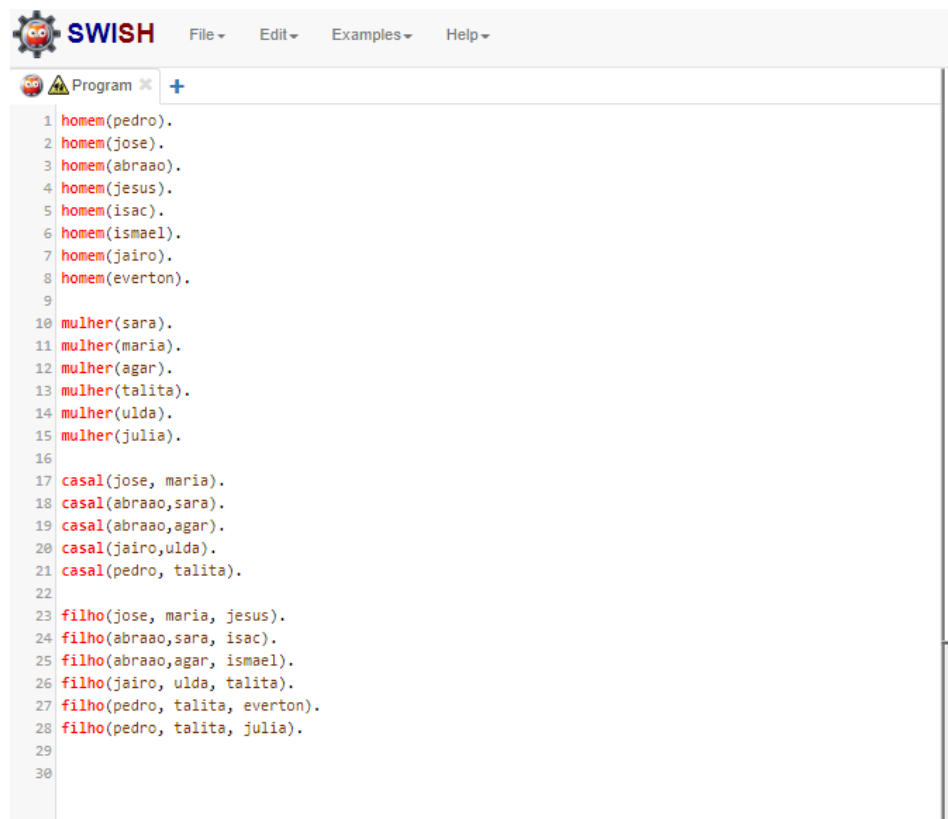
## **Listas (array de elementos)**

Lista é uma das estruturas de uma seqüência ordenada de elementos, muito comum em programação não numérica

Exemplo:

Lista = [a, b, c].

## 2) Utilizando o PROLOG, construa a seguinte Base de Fatos:



```
1 homem(pedro).
2 homem(jose).
3 homem(abraao).
4 homem(jesus).
5 homem(isac).
6 homem(ismael).
7 homem(jairo).
8 homem(everton).
9
10 mulher(sara).
11 mulher(maria).
12 mulher(agar).
13 mulher(talita).
14 mulher(ulda).
15 mulher(julia).
16
17 casal(jose, maria).
18 casal(abraao,sara).
19 casal(abraao,agar).
20 casal(jairo,ulda).
21 casal(pedro, talita).
22
23 filho(jose, maria, jesus).
24 filho(abraao,sara, isac).
25 filho(abraao,agar, ismael).
26 filho(jairo, ulda, talita).
27 filho(pedro, talita, everton).
28 filho(pedro, talita, julia).
29
30
```

Figura 1 - interface Swi Prolog com base de conhecimento (Autoria própria)

## 3) Testar, e registrar as as consultas para:

- a) todas as mulheres;  
inferência feita: **mulher(X)**.

retorno:

**X = sara**

**X = maria**

**X = agar**

**X = talita**

**X = ulda**

**X = julia**

- b) Se existe mulher com nome de maria;  
inferência feita: **mulher(maria)**.

retorno:

true

- c) todos os homens;  
inferência feita: **homem(X)**.

retorno:

**X = pedro**

**X = jose**

**X = abraao**

**X = jesus**

**X = isac**

**X = ismael**

**X = jairo**

**X = everton**

d) Se existe homem com nome de joao;

inferência feita: *homem(joao)*..

retorno:

**false**

e) todos os maridos;

Antes dessa inferência declaramos mais duas regras na base de dados:

marido(X) :-casal(X,\_).

esposa(Y) :-casal(\_,Y).

inferência feita: *marido(X)*.

**X = jose**

**X = abraao**

**X = abraao**

**X = jairo**

**X = pedro**

f) todas esposas;

Antes dessa inferência declaramos mais duas regras na base de dados:

marido(X) :-casal(X,\_).

esposa(Y) :-casal(\_,Y).

inferência feita: *esposa(X)*.

retorno:

**X = maria**

**X = sara**

**X = agar**

**X = ulda**

**X = talita**

g) todos os netos (geral);

Antes dessa inferência declaramos mais uma nova regra na base de dados:

$\text{netos}(A,X) :- \text{filho}(X, \_, Y), \text{filho}(Y, \_, A); \text{filho}(X, \_, Y), \text{filho}(\_, Y, A);$   
 $\text{filho}(\_, X, Y), \text{filho}(\_, Y, A); \text{filho}(\_, X, Y), \text{filho}(Y, \_, A).$

Dentro da variável A está sendo guardado todos os netos de X, buscados pelas regras lógicas posteriores ao :- .

inferência feita:**netos(X, \_)**

retorno:

**X = everton**

**X = julia**

**X = everton**

**X = julia**

se repete pois Everton e Julia são netos de Jairo e Ulda

h) todas as netas (meninas);

Antes dessa inferência declaramos duas novas regras na base de dados:

$\text{netos}(A,X) :- \text{filho}(X, \_, Y), \text{filho}(Y, \_, A); \text{filho}(X, \_, Y), \text{filho}(\_, Y, A);$   
 $\text{filho}(\_, X, Y), \text{filho}(\_, Y, A); \text{filho}(\_, X, Y), \text{filho}(Y, \_, A).$

Dentro da variável X está sendo guardado todos os avós de A, buscados pelas regras lógicas posteriores ao :- .

$\text{netoF}(X) :- \text{netos}(X, \_), \text{mulher}(X).$

Dentro da variável X está sendo guardado todos os netos que são mulheres.

inferência feita:**netoF(X).**

retorno:

**X = julia**

**X = julia**

se repete pois Júlia é neta de Jairo e Ulda

i) todos os netos (meninos);

Antes dessa inferência declaramos duas novas regras na base de dados:

$\text{netos}(A,X) :- \text{filho}(X, \_, Y), \text{filho}(Y, \_, A); \text{filho}(X, \_, Y), \text{filho}(\_, Y, A);$   
 $\text{filho}(\_, X, Y), \text{filho}(\_, Y, A); \text{filho}(\_, X, Y), \text{filho}(Y, \_, A).$

Dentro da variável X está sendo guardado todos os avós de A, buscados pelas regras lógicas posteriores ao :- .

netoM(X) :- netos(X,\_), homem(X).

Dentro da variável X está sendo guardado todos os netos que são homens.

**inferência feita: *netoM(X)*.**

**retorno:**

**X = everton**

**X = everton**

**se repete pois Everton é neto de Jairo e Ulda**

**j) todos os avós (geral);**

**Antes dessa inferência declaramos mais uma nova regra na base de dados:**

avos(X,A):- filho(X,\_,Y), filho(Y,\_,A); filho(X,\_,Y), filho(\_,Y,A);  
                    filho(\_,X,Y), filho(\_,Y,A); filho(\_,X,Y), filho(Y,\_,A).

Dentro da variável X está sendo guardado todos os avós de A, buscados pelas regras lógicas posteriores ao :- .

**inferência feita: *avos(X,\_)***

**retorno:**

**X = jairo**

**X = jairo**

**X = ulda**

**X = ulda**

**se repete pois Jairo e Ulda são avós de Everton e Julia**

**k) todos os avôs (homens);**

**Antes dessa inferência declaramos mais uma nova regra na base de dados:**

avos(X,A):- filho(X,\_,Y), filho(Y,\_,A); filho(X,\_,Y), filho(\_,Y,A);  
                    filho(\_,X,Y), filho(\_,Y,A); filho(\_,X,Y), filho(Y,\_,A).

Dentro da variável X está sendo guardado todos os avôs de A, buscados pelas regras lógicas posteriores ao :- .

avosM(X) :- avos(X,\_), homem(X).

Dentro da variável X está sendo guardado todos os avôs que são homens.

**inferência feita: *avosM(X)*.**

**retorno:**

**X = jairo**

**X = jairo**

**se repete pois Jairo é avôs de Everton e Julia**

**I) todas as avós (mulheres);**

**Antes dessa inferência declaramos mais uma nova regra na base de dados:**

```
avos(X,A):- filho(X, _, Y), filho(Y, _, A); filho(X, _, Y), filho(_, Y, A);  
           filho(_, X, Y), filho(_, Y, A); filho(_, X, Y), filho(Y, _, A).
```

Dentro da variável X está sendo guardado todos os avós de A, buscados pelas regras lógicas posteriores ao :- .

```
avosF(X) :- avos(X,_),mulher(X).
```

Dentro da variável X está sendo guardado todos as avós que são mulheres.

**inferência feita: avosF(X).**

**retorno:**

**X = ulda**

**X = ulda**

**se repete pois Ulga é avó de Everton e Julia**

**O código final portanto fica:**

```
homem(pedro).  
homem(jose).  
homem(abraao).  
homem(jesus).  
homem(isac).  
homem(ismael).  
homem(jairo).  
homem(everton).
```

```
mulher(sara).  
mulher(maria).  
mulher(agar).  
mulher(talita).  
mulher(ulda).  
mulher(julia).
```

```
casal(jose, maria).  
casal(abraao,sara).  
casal(abraao,agar).  
casal(jairo,ulda).  
casal(pedro, talita).
```

```
filho(jose, maria, jesus).  
filho(abraao,sara, isac).  
filho(abraao,agar, ismael).  
filho(jairo, ulda, talita).  
filho(pedro, talita, everton).
```

filho(pedro, talita, julia).

marido(X) :- casal(X, \_).

esposa(Y) :- casal(\_, Y).

avos(X,A):- filho(X, \_, Y), filho(Y, \_, A); filho(X, \_, Y), filho(\_, Y, A);  
                    filho(\_, X, Y), filho(\_, Y, A); filho(\_, X, Y), filho(Y, \_, A).

avosM(X) :- avos(X, \_), homem(X).

avosF(X) :- avos(X, \_), mulher(X).

netos(A,X) :- filho(X, \_, Y), filho(Y, \_, A); filho(X, \_, Y), filho(\_, Y, A);  
                    filho(\_, X, Y), filho(\_, Y, A); filho(\_, X, Y), filho(Y, \_, A).

netoM(X) :- netos(X, \_), homem(X).

netoF(X) :- netos(X, \_), mulher(X).

## REFERÊNCIAS

FRADE, Maria. Lógica Computacional: PROLOG. *In: LÓGICA: PROLOG*. 1. Departamento de Informática Universidade do Minho, 2006. Disponível em:<<https://www4.di.uminho.pt/~mjf/pub/LC-Prolog.pdf>>. Acesso em: 21 out. 2021.

BARANAUSKAS, José. Sintaxe e Semântica de Programas Prolog Programas. *In: Sintaxe e Semântica de Programas Prolog Programas Prolog: Inteligência Artificial*. Departamento de Física e Matemática – FFCLRP-USP, 2004. Disponível em:<<https://dcm.ffclrp.usp.br/~augusto/teaching/ia/IA-Prolog-Sintaxe-Semantica.pdf>> Acesso em: 21 out. 2021.