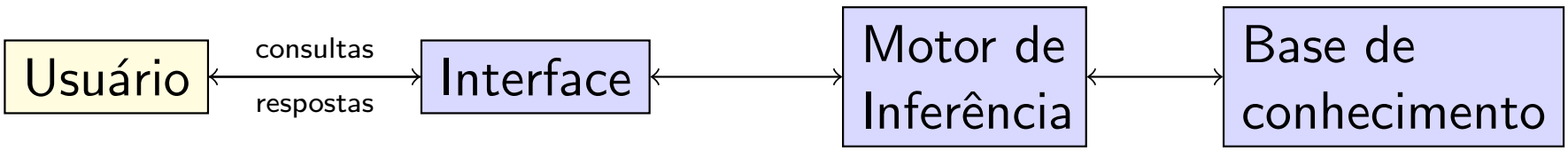
GSI010 - Programação Lógica

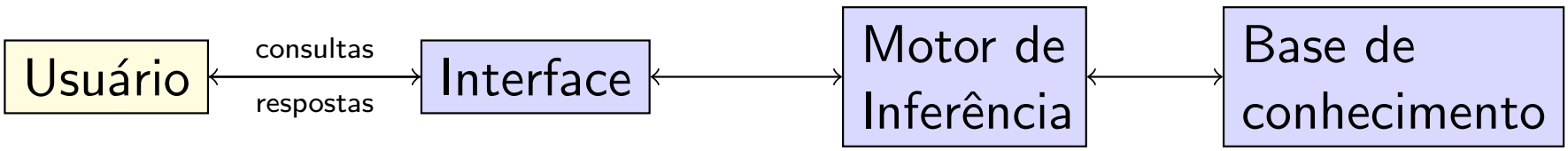
Revisão aula 1



O prolog é baseado em:

- Fatos
  - relações/afirmações entre objetos

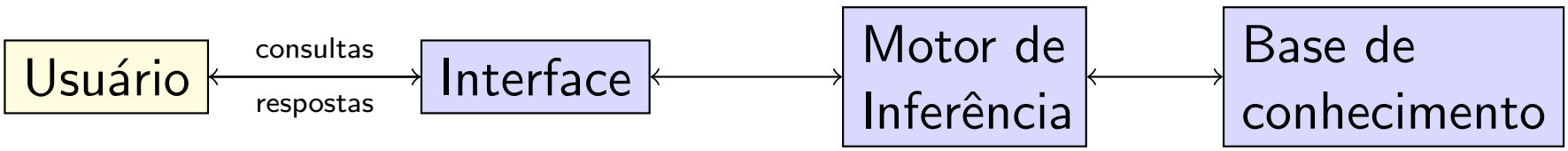
Revisão aula 1



O prolog é baseado em:

- Fatos
  - relações/afirmações entre objetos
- Regras:
  - especificação de algo que pode ser verdadeiro se condições forem satisfeitas

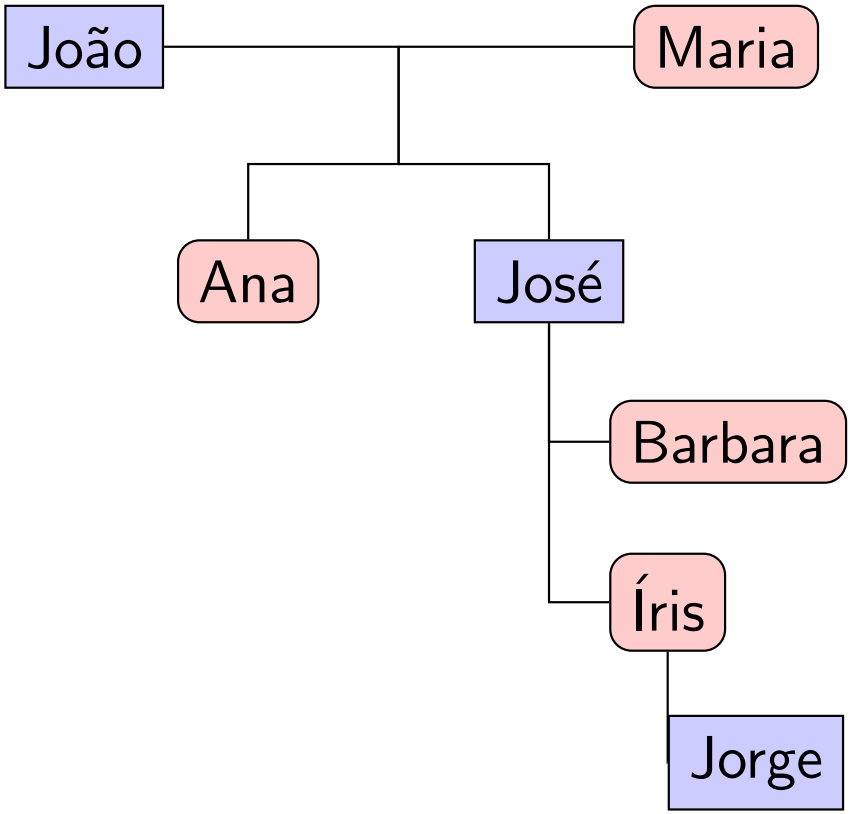
Revisão aula 1



O prolog é baseado em:

- Fatos
  - relações/afirmações entre objetos
- Regras:
  - especificação de algo que pode ser verdadeiro se condições forem satisfeitas
- Consultas:
  - questionamentos que serão respondidos avaliando-se os fatos e regras

Programação em Prolog



Fatos

```
1  progenitor(maria, jose).
2  pai(joao, ana).
3  mae(iris, jorge).
```

Programação em prolog

Regras

- é composta de duas partes: conclusão (esquerda) e condição (direita)
- exemplo: relação Filho.

```
1    avo(Vo,Neto) :- mae(Vo, Mae), mae(Mae, Neto).
```

Programação em prolog

Consultas

- X é uma variável que representa um objeto desconhecido
- variáveis são escritas com a primeira letra em maiúsculo
  - João é filho de José?→ ?− filho(joão, josé).
  - Quem é o filho de José?→?− filho(X, josé).
  - Quem são os filhos de João?→?−filho(X,joão).

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
```

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).
  - cabeça (da regra): filho(X, Y)

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).
  - cabeça (da regra): filho(X, Y)
  - corpo (da regra): progenitor(Y, X)

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).
  - cabeça (da regra): filho(X, Y)
  - corpo (da regra): progenitor(Y, X)
- cláusula: um fato ou uma regra (são 7 nessa base)

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).
  - cabeça (da regra): filho(X, Y)
  - corpo (da regra): progenitor(Y, X)
- cláusula: um fato ou uma regra (são 7 nessa base)
  - marcado por ponto final

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).
  - cabeça (da regra): filho(X, Y)
  - corpo (da regra): progenitor(Y, X)
- cláusula: um fato ou uma regra (são 7 nessa base)
  - marcado por ponto final
- functor: `filho(X,Y)`

Base de conhecimento 2

```
1 progenitor(maria, josé).
2 progenitor(joão, josé).
3 progenitor(joão, ana).
4 progenitor(josé, júlia).
5 progenitor(josé, íris).
6 progenitor(íris, jorge).
7
8 filho(X,Y) :- progenitor(Y,X).
```

Dando nomes:

- fato: progenitor(maria, josé).
- regra: filho(X, Y) :- progenitor(Y, X).
  - cabeça (da regra): filho(X, Y)
  - corpo (da regra): progenitor(Y, X)
- cláusula: um fato ou uma regra (são 7 nessa base)
  - marcado por ponto final
- functor: `filho(X,Y)`
- argumentos: `filho(X, Y)`

Base de conhecimento 3

```
1 feliz(iolanda).
2
```

Base de conhecimento 3

```
1 feliz(iolanda).
2
3 escuta_musica(maria).
4 escuta_musica(iolanda):- feliz(iolanda).
5
```

Base de conhecimento 3

```
1 feliz(iolanda).
2
3 escuta_musica(maria).
4 escuta_musica(iolanda):- feliz(iolanda).
5
6 tocaViolao(maria):- escuta_musica(maria).
7 tocaViolao(iolanda):- escuta_musica(iolanda).
```

São cinco cláusulas na base de conhecimento.

Base de conhecimento 3

```
1 feliz(iolanda).
2
3 escuta_musica(maria).
4 escuta_musica(iolanda):- feliz(iolanda).
5
6 tocaViolao(maria):- escuta_musica(maria).
7 tocaViolao(iolanda):- escuta_musica(iolanda).
```

São cinco cláusulas na base de conhecimento.

São três predicados, `feliz`, `escuta_musica` e ...

```
1 ?- tocaViolao(maria).
2
```

Base de conhecimento 3

```
1 feliz(iolanda).
2
3 escuta_musica(maria).
4 escuta_musica(iolanda):- feliz(iolanda).
5
6 tocaViolao(maria):- escuta_musica(maria).
7 tocaViolao(iolanda):- escuta_musica(iolanda).
```

São cinco cláusulas na base de conhecimento.

São três predicados, `feliz`, `escuta_musica` e ...

```
1 ?- tocaViolao(maria).
2
3 true
4
```

Base de conhecimento 3

```
1 feliz(iolanda).
2
3 escuta_musica(maria).
4 escuta_musica(iolanda):- feliz(iolanda).
5
6 tocaViolao(maria):- escuta_musica(maria).
7 tocaViolao(iolanda):- escuta_musica(iolanda).
```

São cinco cláusulas na base de conhecimento.

São três predicados, `feliz`, `escuta_musica` e ...

```
1 ?- tocaViolao(maria).
2
3 true
4
5 ?- tocaViolao(iolanda).
6
```

Base de conhecimento 3

```
1 feliz(iolanda).
2
3 escuta_musica(maria).
4 escuta_musica(iolanda):- feliz(iolanda).
5
6 tocaViolao(maria):- escuta_musica(maria).
7 tocaViolao(iolanda):- escuta_musica(iolanda).
```

São cinco cláusulas na base de conhecimento.

São três predicados, `feliz`, `escuta_musica` e ...

```
1 ?- tocaViolao(maria).
2
3 true
4
5 ?- tocaViolao(iolanda).
6
7 true
```

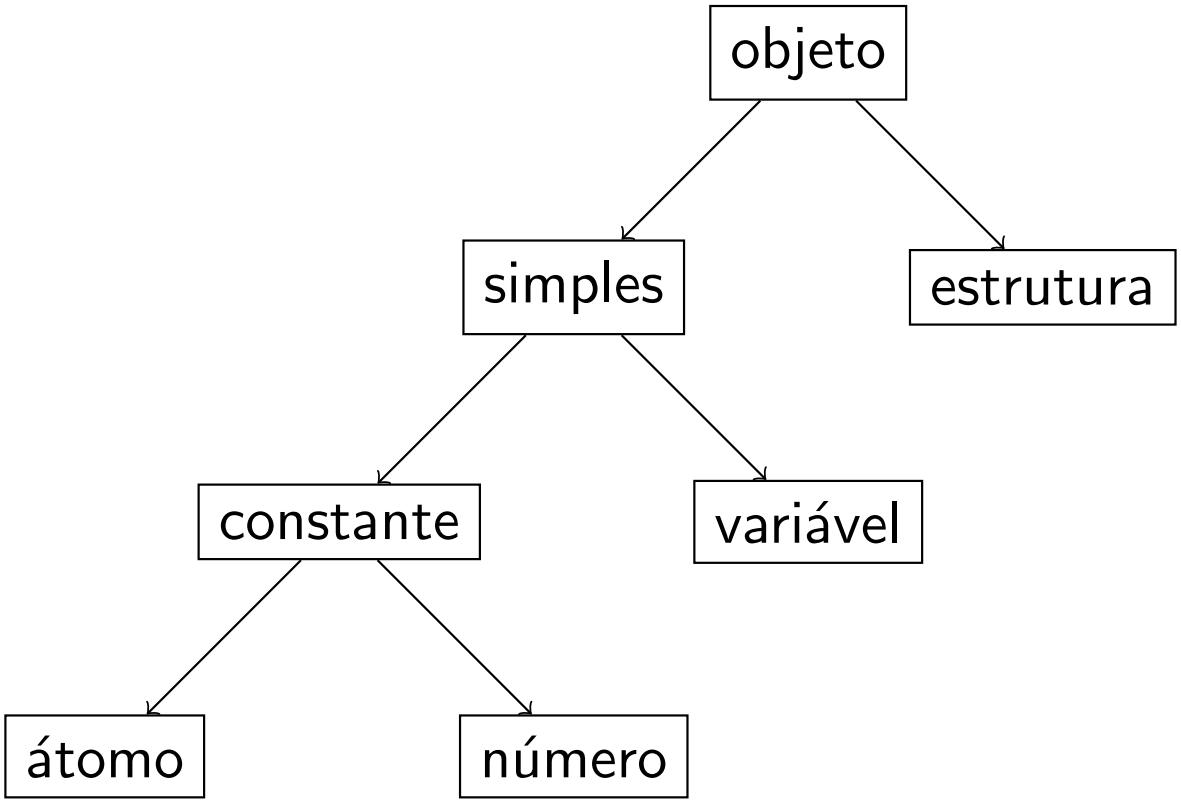
Base de conhecimento 4

```
1 feliz(vitor).
2 escuta_musica(bruno).
3 tocaViolao(vitor):- escuta_musica(vitor),
4                     feliz(vitor).
5 tocaViolao(bruno):- feliz(bruno).
6 tocaViolao(bruno):- escuta_musica(bruno).
7 tocaViolao(bruno):- feliz(bruno); escuta_musica(bruno).
8 tocaViolao(vitor):- \+feliz(maria).
```

Leia-se

- operador "dois pontos hífen" ":-": **se**
- operador "vírgula" ,: "e" (conjunção)
- operador "ponto-e-vírgula" ; : "ou" (disjunção)
- operador "\+": negação

Sintaxe e semântica



Sintaxe

- Átomos podem ser formados por:
  - Cadeia de letras ou dígitos, iniciando obrigatoriamente com letra minúscula. (pode conter o símbolo _)
    - Ex: socrates, joao, x_1
  - Cadeia de caracteres quaisquer, inclusive espaço em branco, desde que usando o símbolo ' (aspas simples)
    - Ex: 'Prog. Lógico'

Sintaxe

- Números
  - Exemplos:
    - 1
    - 123
    - -123
- Reais
  - 3.14159
  - -123.41

Sintaxe

- Variáveis
  - São cadeias de letras, dígitos ou caractere sublinhado (_), devendo iniciar com este ou com uma letra maiúscula
  - O caractere "_", sozinho, representa uma variável anônima, isto é, sem interesse para um determinado procedimento
  - Exemplos de variáveis: X, Resultado, _var, _

Sintaxe

- Variáveis
  - O escopo léxico de nomes de variáveis é apenas uma cláusula
    - Ex: se o nome X25 ocorre em duas cláusulas diferentes, então ele está representando duas variáveis diferentes; toda ocorrência de X25 dentro da mesma cláusula quer significar a mesma variável

Sintaxe

- Variáveis
  - Uma variável pode estar
    - Instanciada: quando á variável já referencia algum objeto
    - não-instanciada: quando a variável não referencia nenhum objeto, ou seja, quando o objeto a que ela referencia ainda não é conhecido
  - Quando uma variável é usada numa pergunta, o Prolog procura todos os fatos tentando encontrar um objeto no qual a variável possa ser instanciada
    - Quando uma solução é encontrada, ela é mostrada. Se o usuário estiver satisfeito com a resposta, basta digitar return
    - Se desejar mais respostas, usa-se ponto-evírgula ";"

Sintaxe

- Variáveis

```
1 gosta(joao, peixe).
2 gosta(joao,maria).
3 gosta(maria,livro).
4 gosta(pedro,livro).
5 gosta(maria,flor).
6 gosta(maria,vinho).
```

Qual o resultado das seguintes perguntas?

```
1 ?- gosta(maria,X).
2 ?- gosta(X,livro).
3 ?- gosta(Quem,Oque).
4 ?- gosta(X,Y).
5 ?- gosta(X,X).
6 ?- gosta(_a,_b).
7 ?- gosta(A,peixe).
```

Sintaxe

- Estruturas
  - São objetos que possuem vários componentes
  - Os próprios componentes podem ser também estruturas
  - Ex: Data $\rightarrow$ estrutura com 3 componentes
    - data(1, maio, 2013).
    - functor data
    - argumentos (1, maio, 2013)

Sintaxe

- Estruturas
  - Ex: data(Dia, março, 1996)
    - A variável Dia pode ser unificada para qualquer objeto
  - Todos os objetos em Prolog são denominados termos.
    - Toda constante é um termo
    - Toda variável é um termo
    - Se $t1, t2, \ldots, tn$ são termos, então $f(t1, t2, ..., tn)$ também é um termo

Sintaxe

- Estruturas
  - Todos os objetos estruturados podem ser representados como árvores
    - A raiz da árvore é o functor
    - os ramos que dela partem são os argumentos ou componentes
    - Ex: (a + b) * (c - 5)
    - *(+(a, b), -(c, 5))

Sintaxe

- Unificação
  - É o processo que, dados dois termos como dados de entrada verifica-se eles se "casam"
  - Dados dois termos, diz-se que eles unificam se
    - Eles são idênticos ou
    - As variáveis de ambos os termos podem ser instanciadas com objetos de maneira que, após a substituição das variáveis por esses objetos, os termos se tornam idênticos

Sintaxe

- Unificação
  - Isto significa que:
    - maria e maria se unificam
    - 42 e 42 se unificam
    - mulher(maria) e mulher(maria) se unificam
  - Isto também significa que:
    - vitor e maria não se unificam
    - mulher(maria) e mulher(joana) não se unificam

Sintaxe

- Unificação
- Os termos abaixo se unificam?
  - maria e X

Sintaxe

- Unificação
- Os termos abaixo se unificam?
  - maria e X sim
  - mulher(Z) e mulher(maria)

Sintaxe

- Unificação
- Os termos abaixo se unificam?
  - maria e X sim
  - mulher(Z) e mulher(maria) sim Z = maria
  - ama(maria,X) e ama(X,vitor)

Sintaxe

- Unificação
- Os termos abaixo se unificam?
  - maria e X sim
  - mulher(Z) e mulher(maria) sim Z = maria
  - ama(maria,X) e ama(X,vitor) não

Sintaxe

- Unificação
- Os termos abaixo se unificam?
  - maria e X sim
  - mulher(Z) e mulher(maria) sim Z = maria
  - ama(maria,X) e ama(X,vitor) não

Falha

Se os termos não unificam, dizemos que o processo falha. Caso contrário, é bem-sucedido

Unificação

```
1 ?- data(D, M, 1994) = data(X, março, A).
```

Unificação

```
1 ?- data(D, M, 1994) = data(X, março, A).
2 D = X,
3 M = 'março',
4 A = 1994.
5
```

Unificação

```
1 ?- data(D, M, 1994) = data(X, março, A).
2 D = X,
3 M = 'março',
4 A = 1994.
5
6 ?- data(D, M, 1994) = data(X, Y, 94).
```

Unificação

```
1 ?- data(D, M, 1994) = data(X, março, A).
2 D = X,
3 M = 'março',
4 A = 1994.
5
6 ?- data(D, M, 1994) = data(X, Y, 94).
7 false.
8
```

Unificação

```
1 ?- data(D, M, 1994) = data(X, março, A).
2 D = X,
3 M = 'março',
4 A = 1994.
5
6 ?- data(D, M, 1994) = data(X, Y, 94).
7 false.
8
9 ?- triângulo(ponto(1, 1), A, ponto(2, 3)) = triângulo(X
      , ponto(4, Y), ponto(2,Z)).
```

Unificação

```
1  ?- data(D, M, 1994) = data(X, março, A).
2  D = X,
3  M = 'março',
4  A = 1994.
5
6  ?- data(D, M, 1994) = data(X, Y, 94).
7  false.
8
9  ?- triângulo(ponto(1, 1), A, ponto(2, 3)) = triângulo(X
       , ponto(4, Y), ponto(2,Z)).
10 A = ponto(4, Y),
11 X = ponto(1, 1),
12 Z = 3.
```

Consulta

```
1  grande(urso).
2  grande(elefante).
3  pequeno(gato).
4  marrom(urso).
5  preto(gato)
6  cinza(elefante).
7
8  escuro(Z) :-
9      preto(Z).
10 escuro(Z) :-
11     marrom(Z).
```

Qual o resultado da consulta:

```
1  ?- escuro(X), grande(X).
```

Consulta

```
1  grande(urso).
2  grande(elefante).
3  pequeno(gato).
4  marrom(urso).
5  preto(gato)
6  cinza(elefante).
7
8  escuro(Z) :-
9      preto(Z).
10 escuro(Z) :-
11     marrom(Z).
```

Qual o resultado da consulta:

```
1 ?- escuro(X), grande(X).
2 X = urso.
```

Exemplos

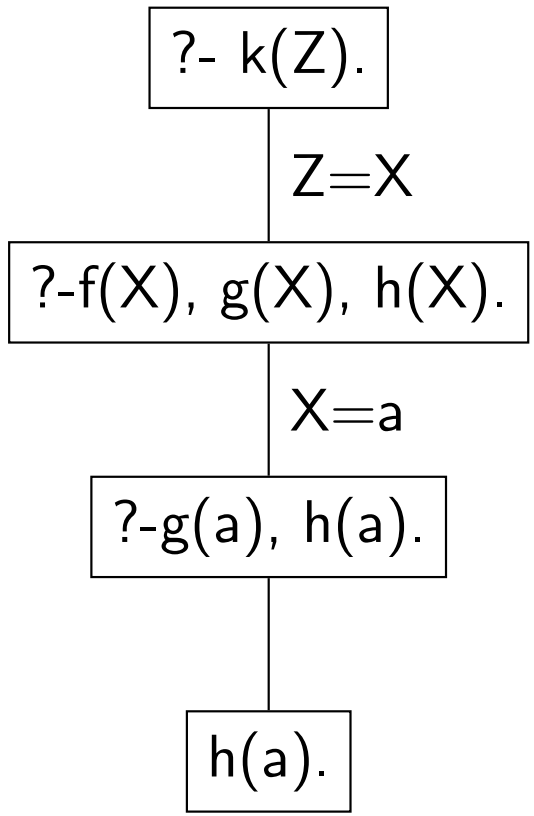
```
1 ?- k(s(g),Y) = k(X,t(k)).
2
3 ?- k(s(g),t(k)) = k(X,t(Y)).
```

Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 f(a).
2 f(b).
3 g(a).
4 g(b).
5 h(b).
6 k(X):- f(X), g(X), h(X).
```

```
1 ?- k(Z).
```

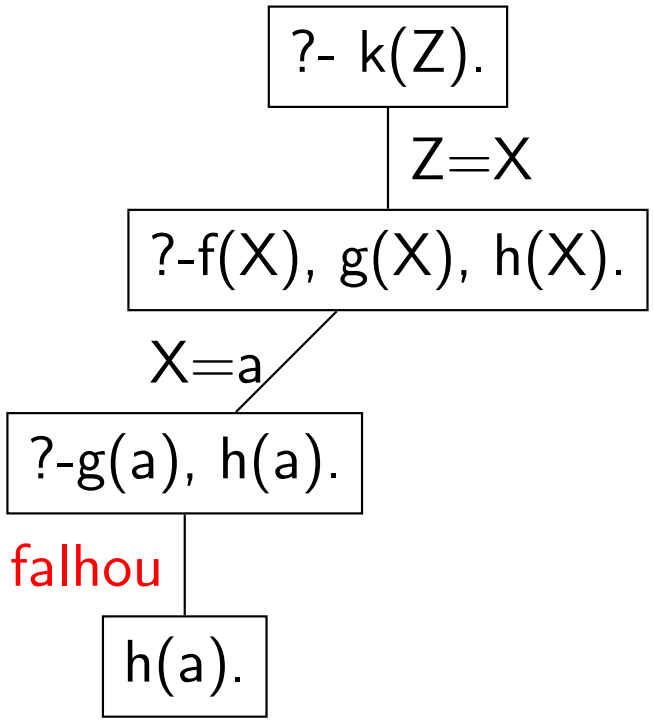


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 f(a).
2 f(b).
3 g(a).
4 g(b).
5 h(b).
6 k(X):- f(X), g(X), h(X).
```

```
1 ?- k(Z).
2 Z=b.
```

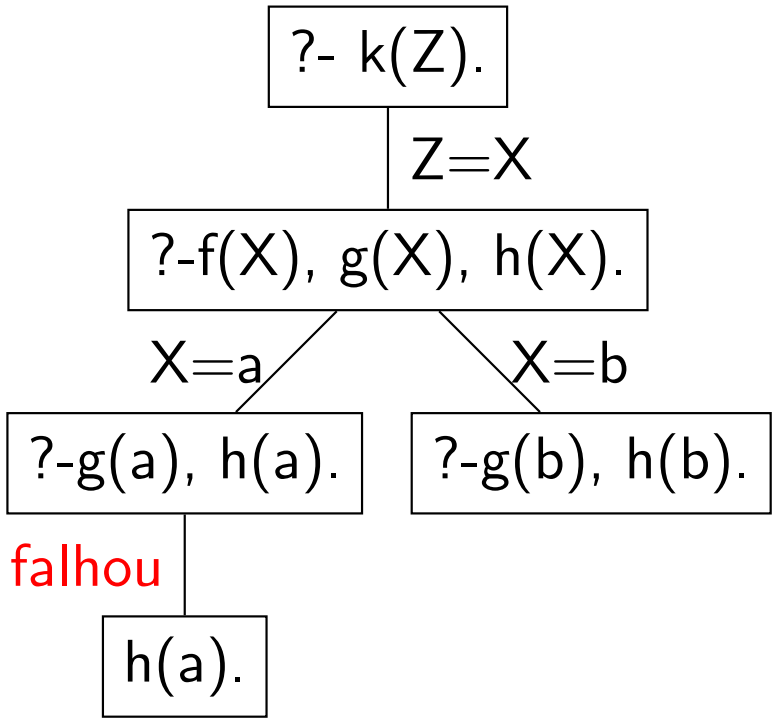


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 f(a).
2 f(b).
3 g(a).
4 g(b).
5 h(b).
6 k(X):- f(X), g(X), h(X).
```

```
1 ?- k(Z).
2 Z=b.
```

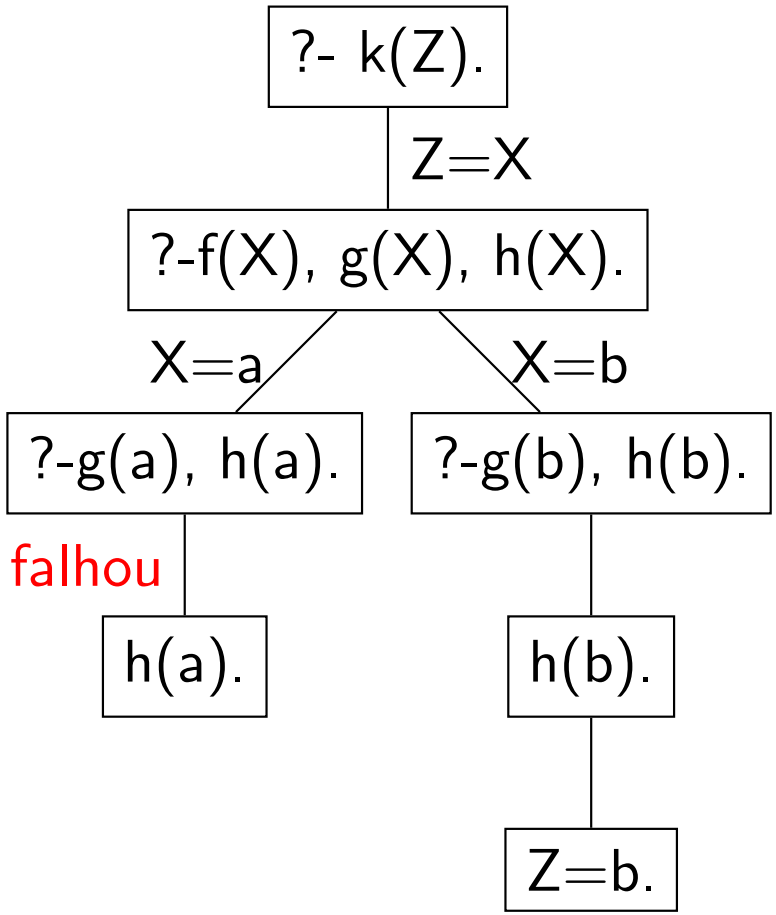


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 f(a).
2 f(b).
3 g(a).
4 g(b).
5 h(b).
6 k(X):- f(X), g(X), h(X).
```

```
1 ?- k(Z).
2 Z=b.
```



Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

?- tem_ciume(X,Y).

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

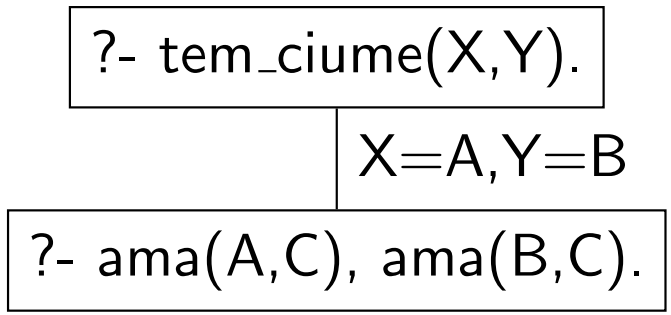
```
1 ?- tem_ciume(X,Y).
```

Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

```
1 ?- tem_ciume(X,Y).
```

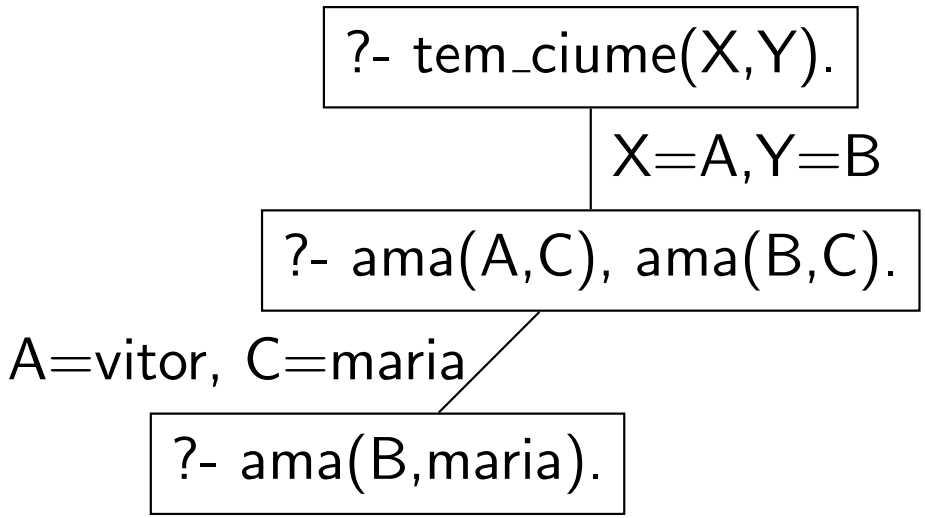


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

```
1 ?- tem_ciume(X,Y).
```

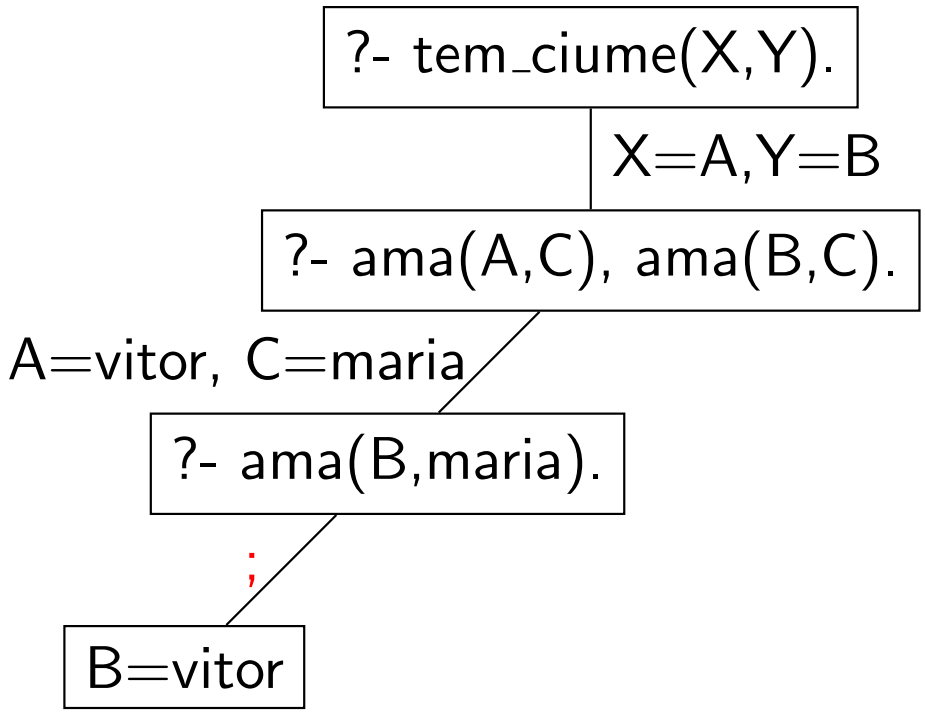


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

```
1 ?- tem_ciume(X,Y).
```

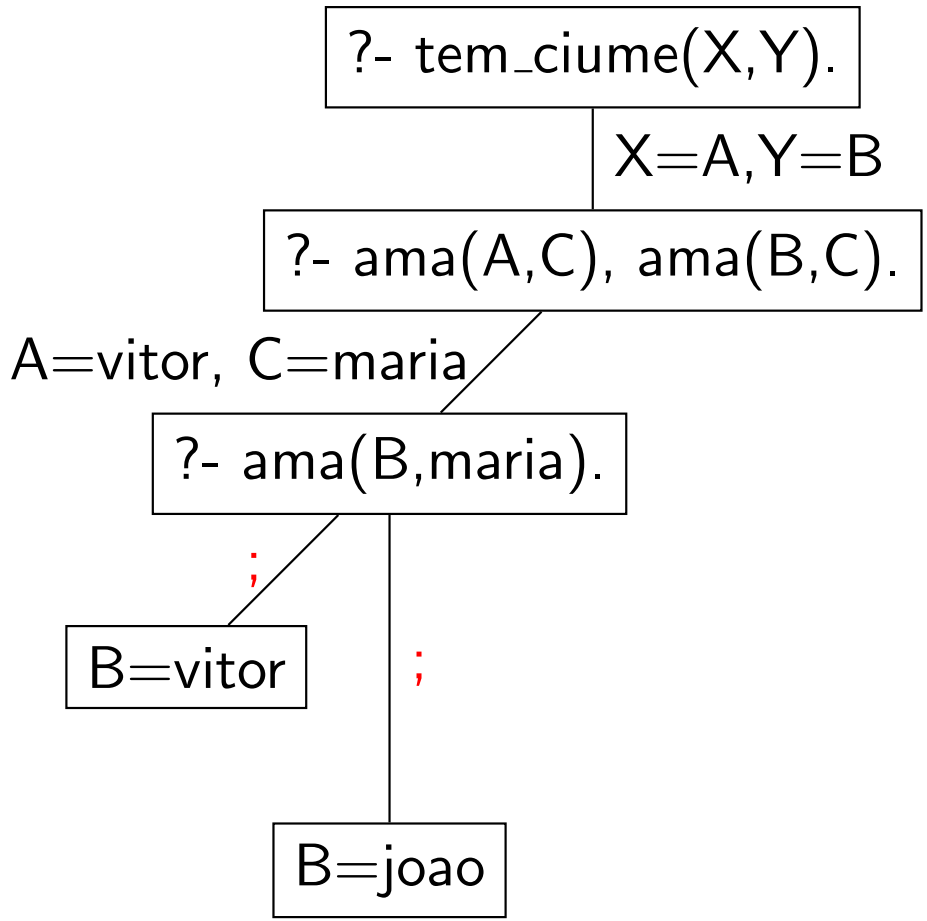


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

```
1 ?- tem_ciume(X,Y).
```

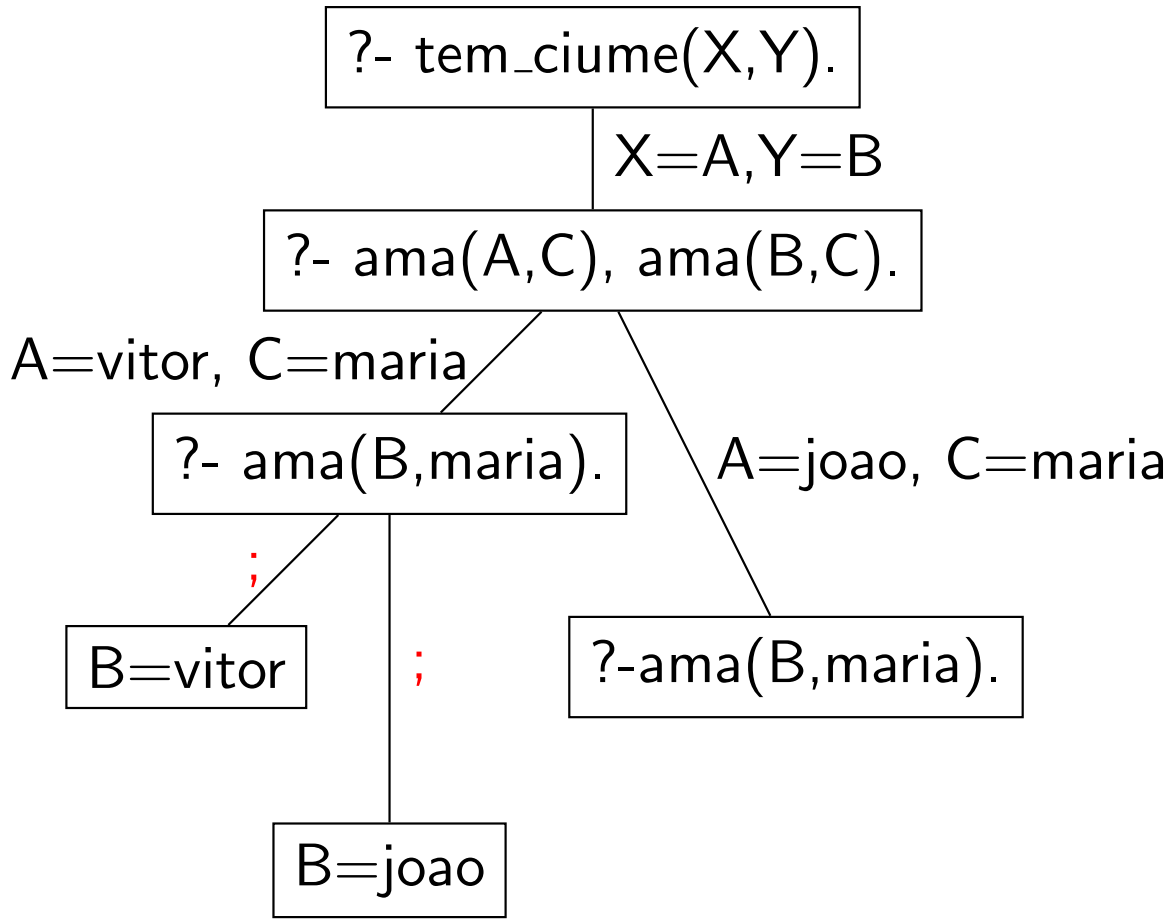


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

```
1 ?- tem_ciume(X,Y).
```

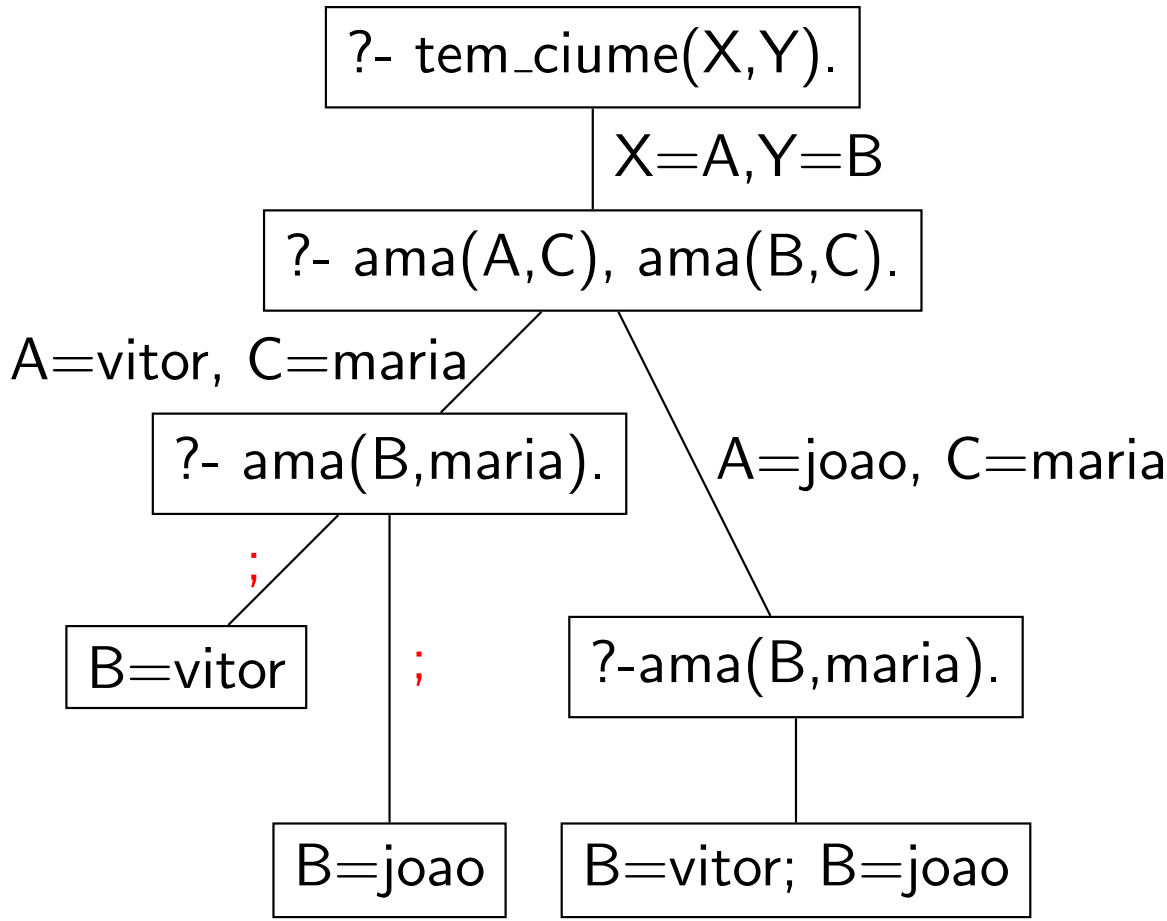


Busca pela Prova

Começaremos a aprender sobre a busca pela prova:

```
1 ama(vitor,maria).
2 ama(joao,maria).
3 tem_ciume(A,B):-
4 ama(A,C),
5 ama(B,C).
```

```
1 ?- tem_ciume(X,Y).
```



Referências

- Luis, A. M. Palazzo, Introdução à programação prolog, Educat, 1997
- Slides profs. Elaine Faria, Hiran Nonato e Gabriel Coutinho - UFU
- Slides da Profa. Solange - ICMC - USP