



Paradigma Lógico

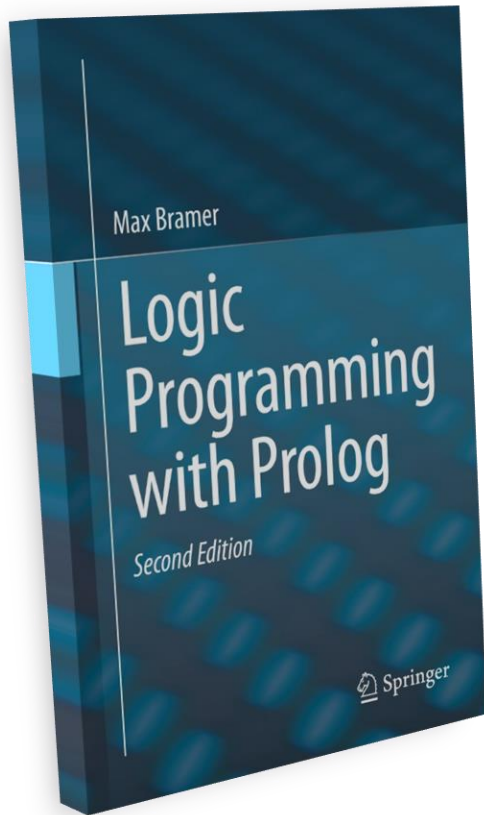
Introdução à Linguagem PROLOG

PROgramação em LÓGica

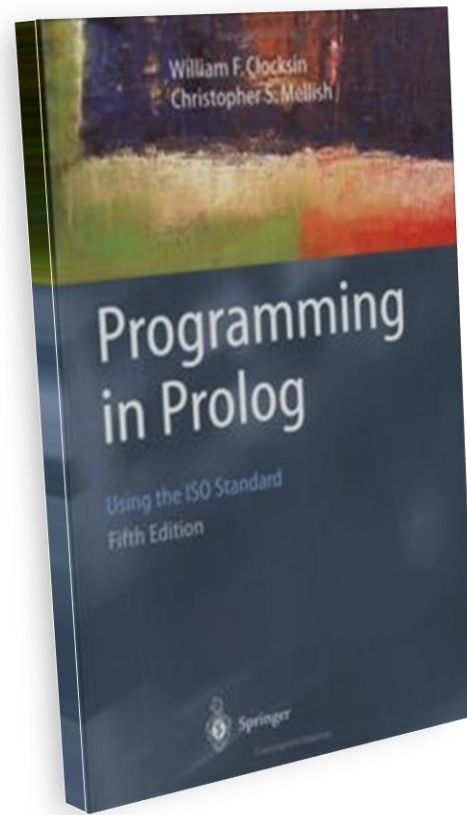


Prof. Ausberto S. Castro V.
ascv@uenf.br

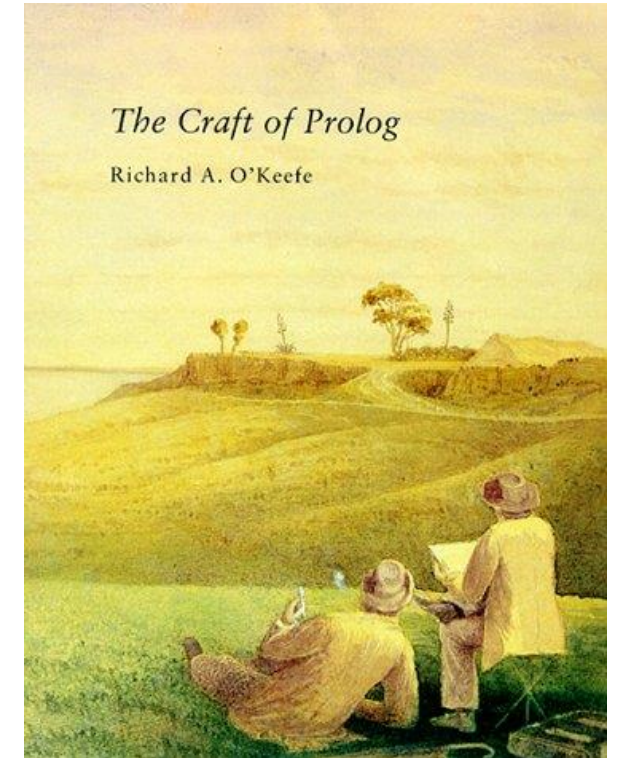
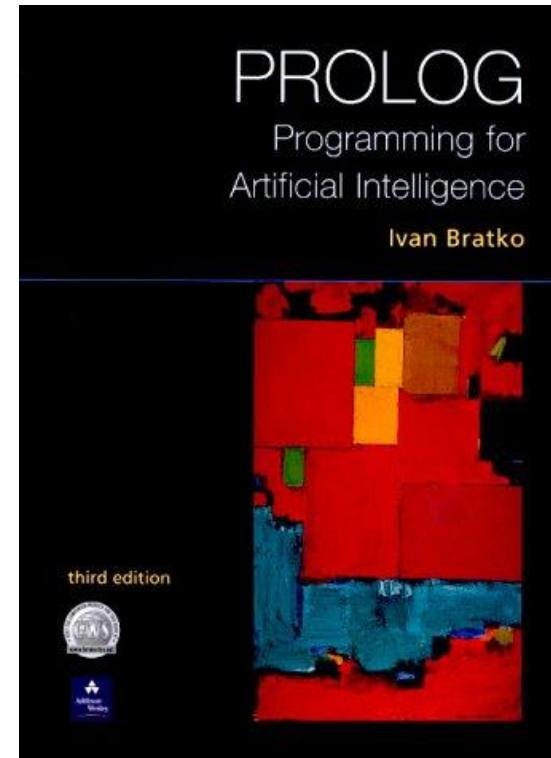
Bibliografia



Max Bramer



W.Clocksin



Lógica

- ❖ **LÓGICA: estudo das proposições**

- ❖ **Conjunção** p AND q

- $p \wedge q$

- ❖ **Disjunção** p OR q

- $p \vee q$

- ❖ **Cláusula de Horn (argumento)**

 - É uma disjunção de literais com pelo menos um literal positivo

$$\neg p \vee \neg q \vee \dots \vee \neg t \vee u$$

ou equivalentemente

$$(p \wedge q \wedge \dots \wedge t) \rightarrow u$$

- ❖ **Predicado**

 - Uma função $P(p,q,r,\dots)$ que é verdadeiro ou falso
 - Uma proposição composta

Lógica - Tabelas-verdade

V = verdadeiro
F = falso

Conjunção		
p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Disjunção		
p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

Negação	
p	$\neg p$
V	F
F	V

$$p \rightarrow q \equiv \neg p \vee q$$

Condicional

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

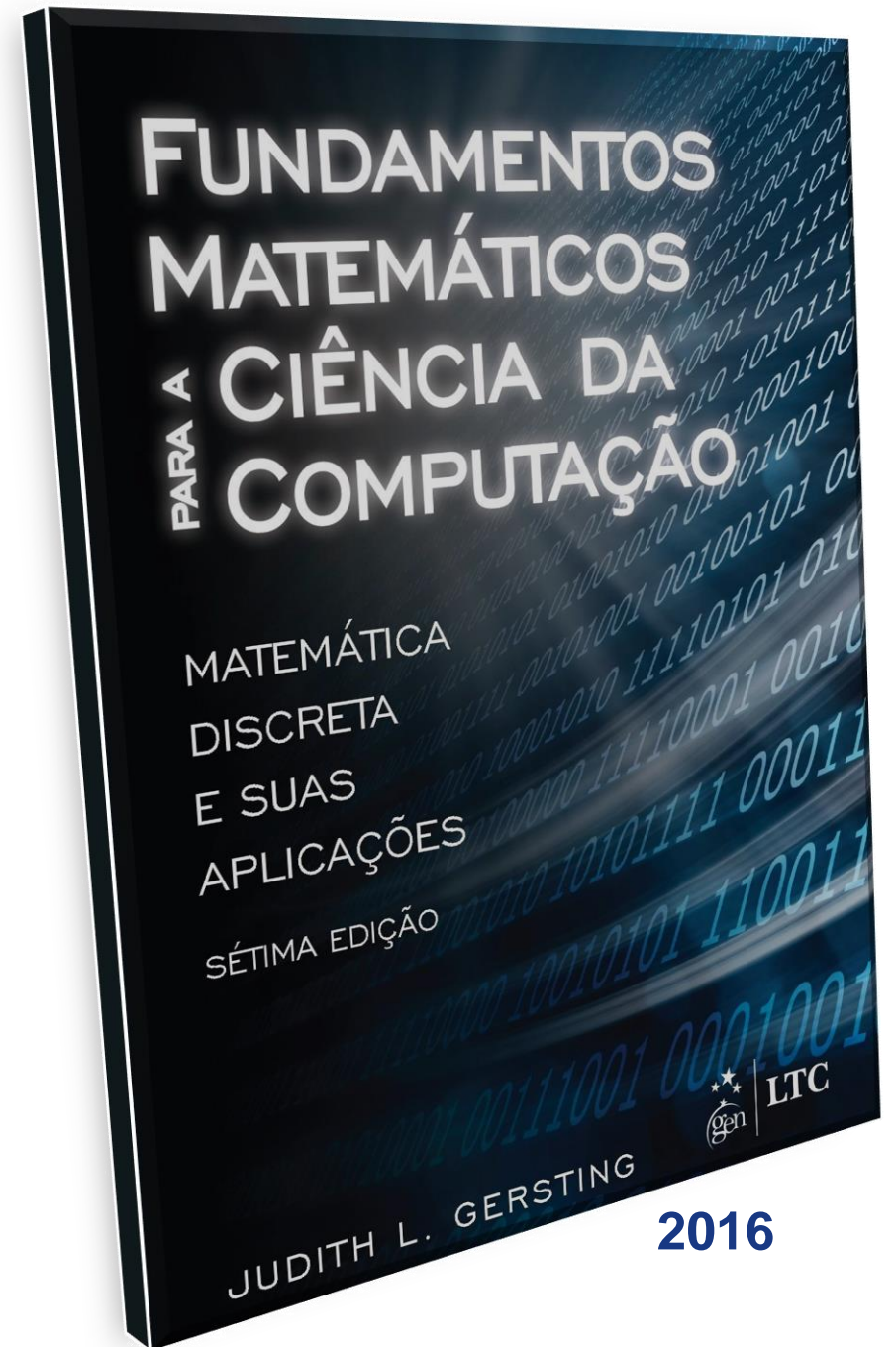
BiCondicional

Cálculo Proposicional

Você foi convocado a participar do júri em um processo criminal. O advogado de defesa argumenta o seguinte:

- Se meu cliente fosse culpado, a faca estaria na gaveta. Ou a faca não estava na gaveta ou Jason Pritchard viu a faca. Se a faca não estava lá no dia 10 de outubro, segue que Jason Pritchard não viu a faca. Além disso, se a faca estava lá no dia 10 de outubro, então a faca estava na gaveta e o martelo estava no celeiro. Mas todos nós sabemos que o martelo não estava no celeiro. Portanto, senhoras e senhores do júri, *meu cliente é inocente*.

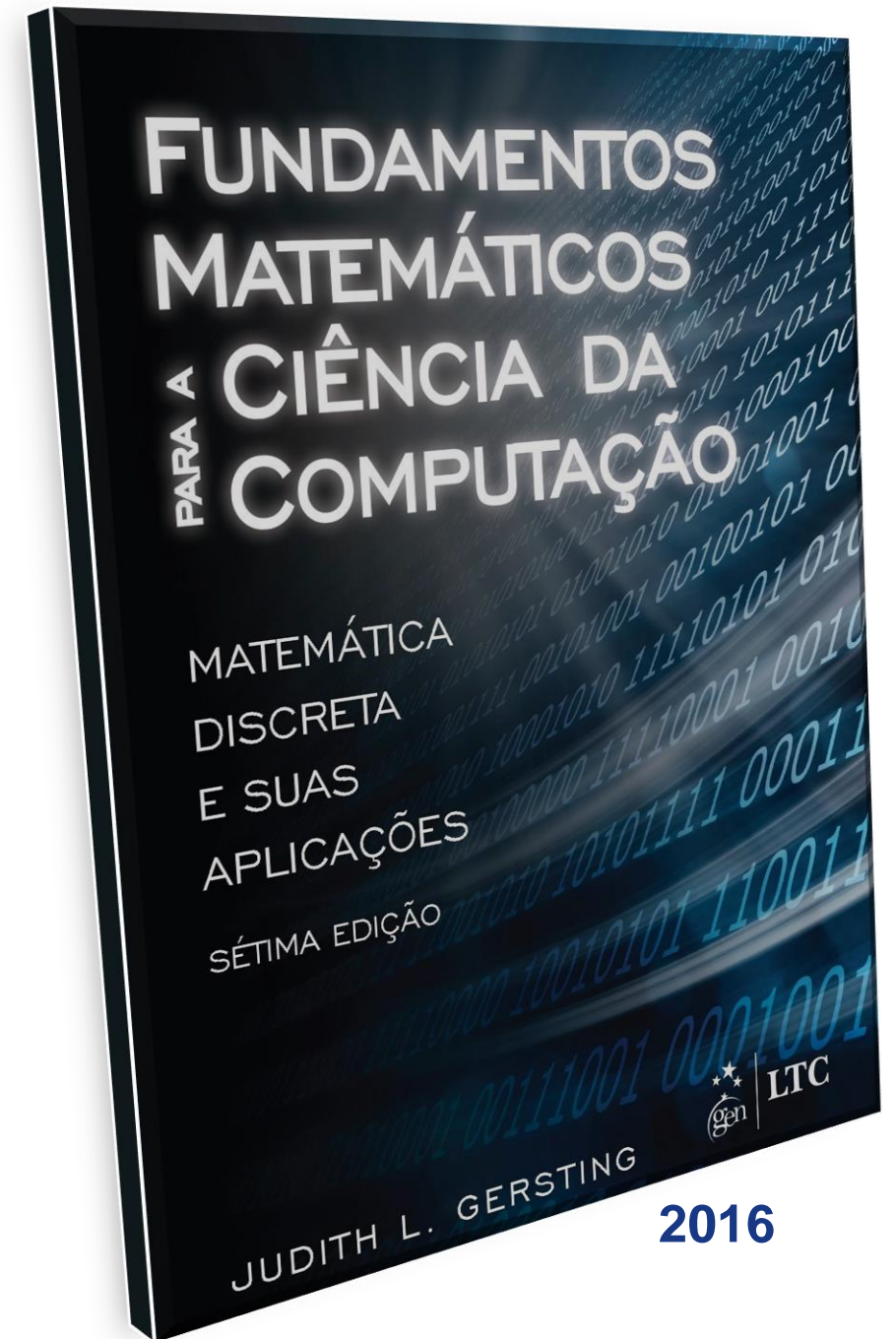
PERGUNTA: O argumento do advogado está correto? Como você deveria votar?



Você foi convocado a participar do júri em um processo criminal. O advogado de defesa argumenta o seguinte:

- z = meu cliente é culpado
- q = a faca estaria na gaveta.
- r = Jason Pritchard viu a faca
- u = martelo estava no celeiro

- $z \rightarrow q$
- $t = q \vee r$
- $\sim q \rightarrow \sim r$
- $q \rightarrow q \wedge u$
- $\sim u$
- $\therefore \sim z$



2016

Argumento válido

- ❖ Um **argumento** é uma sequência (conjunção) de sentenças chamadas *premissas*, seguida de outra sentença chamada *conclusão*, na forma de uma implicação:

$$p_1 \wedge p_2 \wedge p_3 \wedge \dots \wedge p_n \rightarrow q$$

- ❖ Se um argumento

$$p_1 \wedge p_2 \wedge p_3 \wedge \dots \wedge p_n \rightarrow q$$

é uma tautologia, então é chamado de *argumento válido*

- Se $(p_1 \wedge p_2) \rightarrow q$ é um argumento válido, então dizemos que a conclusão q é deduzida ou inferida da verdade das premissas p_1 e p_2
- ❖ Argumentos não válidos são chamados de *falacias*

Argumento válido

$$p_1 \wedge p_2 \wedge p_3 \wedge \dots \wedge p_n \rightarrow q$$

p_1 and p_2 and p_3 and ... and $p_n \rightarrow q$

Se p_1 and p_2 and p_3 and ... and p_n
então q

Se q então p_1 and p_2 and p_3 and ...
and p_n

q :- p_1 , p_2 , p_3 , ... , p_n

Implicação Lógica

- ❖ Se p e q são duas proposições qualquer tais que $p \rightarrow q$ é uma tautologia, então dizemos que p *implica logicamente* q e escrevemos $p \Rightarrow q$
 - Neste caso nos referimos a $p \rightarrow q$ como uma *implicação lógica*
 - Se $p \Rightarrow q$ e $q \Rightarrow p$ então escrevemos $p \Leftrightarrow q$

Regras de Inferência

- ❖ **Permitem a dedução de novas proposições a partir de proposições anteriores**

fbf = fórmula bem formada: p , $p \wedge q$, $p \vee q$, $\sim p$

- ❖ **Permitem a dedução de novas fbf a partir de fbf anteriores**

- **Pedro lê jornal**
- **Pedro lê revistas**

Então Pedro lê jornal e revistas

Regras de Inferência

❖ Modus Ponens MP

$$\frac{p, p \rightarrow q}{\therefore q}$$

❖ Modus Tollens MT

$$\frac{p \rightarrow q, \sim q}{\therefore \sim p}$$

❖ Silogismos HS

$$\frac{p \rightarrow q, q \rightarrow r}{\therefore p \rightarrow r}$$

❖ Conjunção CONJ

$$\frac{p, q}{\therefore p \wedge q}$$

❖ Simplificação SIMP

$$\frac{p \wedge q}{\therefore p} \quad \frac{p \wedge q}{\therefore q}$$

❖ Dilema DC

$$\frac{p \rightarrow q, r \rightarrow s, p \vee r}{\therefore q \vee s}$$

❖ Silogismo Disjuntivo SD

$$\frac{p \vee q, \sim p}{\therefore q} \quad \frac{p \vee q, \sim q}{\therefore p}$$

❖ Adição ADD

$$\frac{p}{\therefore p \vee q} \quad \frac{q}{\therefore p \vee q}$$

Regras de Inferência

❖ **Amplificação AMP**

$$\frac{p}{\therefore p \vee q}$$

❖ **Prova Condicional CON**

$$\frac{\begin{array}{c} p \quad q \\ p \rightarrow (q \rightarrow r) \end{array}}{\therefore r}$$

❖ **Prova por Casos CAS**

$$\frac{\begin{array}{c} p \rightarrow r \\ q \rightarrow r \end{array}}{\therefore (p \vee q) \rightarrow r}$$

❖ **Dilema Destrutivo DD**

$$\frac{\begin{array}{c} p \rightarrow q \\ r \rightarrow s \\ \sim q \vee \sim s \end{array}}{\therefore \sim p \vee \sim r}$$

Modus Ponens MP



- **Maria ganha um milhão de reais**
- **Se Maria ganha um milhão de reais então Luisa abandona o emprego**
- **Portanto, Luisa abandona o emprego**



- **Se João vai a Fortaleza de férias, então ele ganha um desconto escolar**
- **João já está de férias em Fortaleza**
- **Portanto, João já ganhou um desconto escolar**

Modus Tollens MT



- Se Carlos for eleito Presidente, então Guilherme será nomeado como ministro de Cultura
- Guilherme não é o ministro de Cultura
- Portanto, Carlos não foi eleito Presidente



- Se tiver chuva nas quartas-feiras então teremos aula teórica
- Nesta quarta tivemos aula prática
- Portanto, não teve chuva nesta quarta-feira

Exemplo



- Suzana esta preparando um bolo p
- Se Suzana esta preparando um bolo, então ela não esta dando aulas de piano $p \rightarrow \sim q$
- Se Suzana não esta dando aulas de piano, então ela não pode pagar o seguro do seu carro $\sim q \rightarrow \sim r$
- Por tanto, Suzana não pode pagar o seguro do seu carro $\sim r$

$$\begin{array}{c} p \\ p \rightarrow \sim q \\ \sim q \rightarrow \sim r \\ \hline \therefore \sim r \end{array}$$

□ Demonstrar a validade do argumento

Demonstrando a validade do argumento

❖ Passos

1. $p \rightarrow \sim q$
2. $\sim q \rightarrow \sim r$
3. $p \rightarrow \sim r$
4. p
5. $\therefore \sim r$

Razões

premissa
premissa
1. e 2. e HS
premissa
4. e MP

❖ Outra forma

1. p
2. $p \rightarrow \sim q$
3. $\sim q$
4. $\sim q \rightarrow \sim r$
5. $\therefore \sim r$

premissa
premissa
1. e 2. e MP
premissa
3. e 4. e MP

Exercícios

❖ Demonstrar a validade do argumento

$$\begin{array}{l} p \rightarrow r \\ r \rightarrow s \\ t \vee \sim s \\ \sim t \vee u \\ \hline \sim u \\ \hline \therefore \sim p \end{array}$$



1. $p \rightarrow r, r \rightarrow s$ premissa
2. $p \rightarrow s$ 1. e HS
3. $t \vee \sim s$ premissa
4. $\sim s \vee t$ 3.
5. $s \rightarrow t$ 4.
6. $p \rightarrow t$ 2. e 5.
7. $\sim t \vee u$ premissa
8. $t \rightarrow u$ 7.
9. $p \rightarrow u$ 6. e 8.
10. $\sim u$ premissa
11. $\sim p$ 9. e 10.

Exercícios - Provar os seguintes argumentos:

$$\begin{array}{l} p \rightarrow r \\ \sim p \rightarrow q \\ q \rightarrow s \\ \hline \therefore \sim r \rightarrow s \end{array}$$
$$\begin{array}{l} p \rightarrow q \\ q \rightarrow (r \wedge s) \\ \sim r \vee (\sim t \vee u) \\ p \wedge t \\ \hline \therefore u \end{array}$$
$$\begin{array}{l} \sim p \rightarrow q \\ q \rightarrow r \\ \sim r \\ \hline \therefore p \end{array}$$

História do Prolog

❖ 1974 – Bob Kowalski

- University of Edinburgh, Escócia
- Demonstração automatizada de teoremas
- “Predicate Logic as Programming Language”
- IFIP Congress, Stockholm

❖ 1975 - Alain Colmerauer & Philippe Roussel

- Interpreter for the first Logic Programming Language – Prolog
 - University of Aix-Marseille, França, 1972
 - Processamento de linguagem natural

❖ Outros

- Fifth Generation Computing Systems (FGCS)
 - Tokyo, 1981
- Turbo Prolog, Borland, 1985

Paradigma PROLOG

❖ PROLOG:

- PROgramming in LOGic

❖ PROLOG é uma linguagem:

- Declarativa
 - especifica (declara, descreve, afirma) fatos e relacionamentos lógicos
- Simbólica
 - símbolos são utilizados para representar objetos
- Alto nível:
 - Contém um mecanismo embutido (dedução) para resolver problemas (pensar??)

Paradigma PROLOG

❖ PROLOG:

- Prolog é uma linguagem de programação para computação simbólica não-numérica

❖ Programas PROLOG

- Construídos para resolver problemas que incluem OBJETOS e relacionamentos entre objetos
- Resolvem problemas declarando **objetos** e seus **relacionamentos**



Paradigma PROLOG

❖ Exemplo:

- Sergio é proprietário de um livro

❖ Relacionamento

- proprietário

❖ Objetos

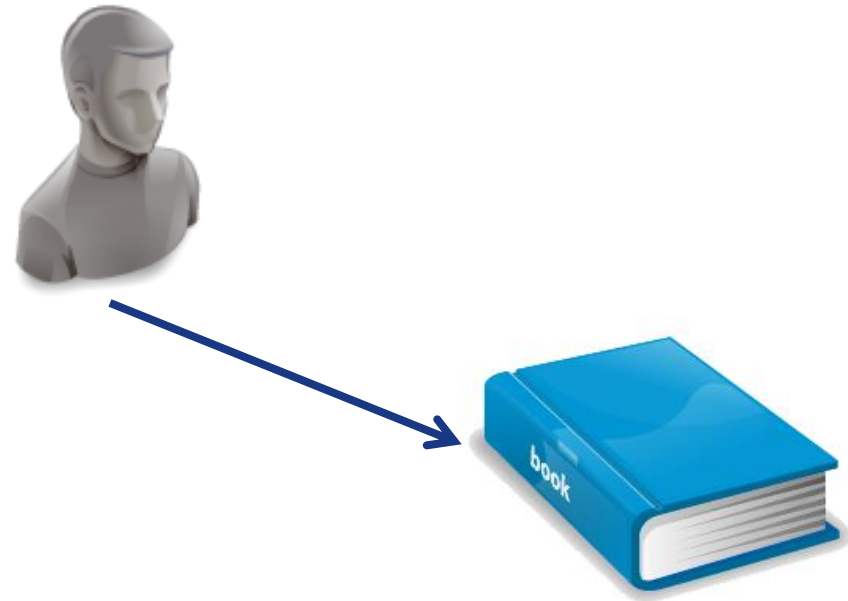
- Sergio
- Livro

❖ Direcional

- Sergio é proprietário de um livro
- Porém, o livro não é proprietário de Sergio

❖ Consultas

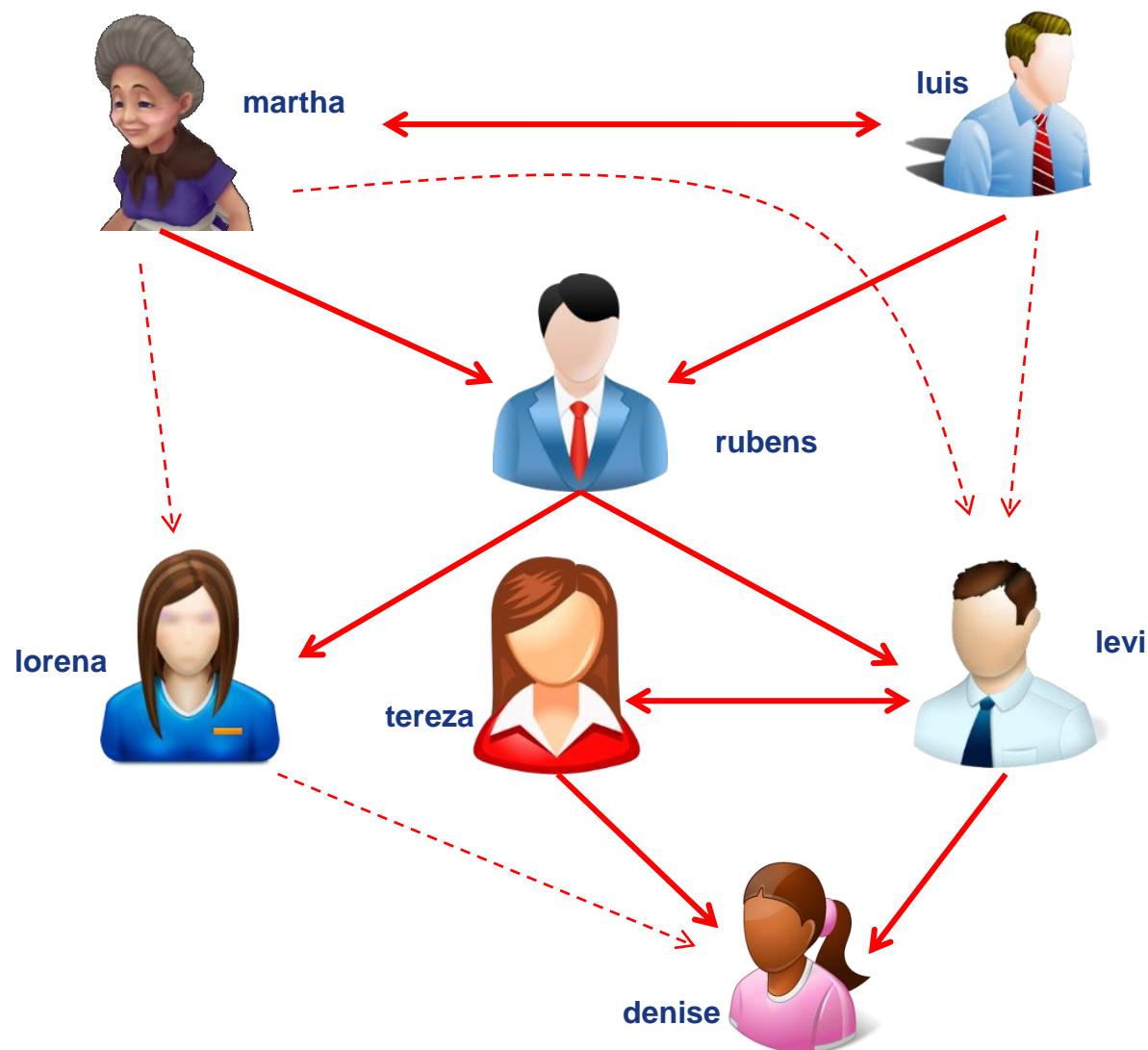
- Sergio é proprietário do livro? (Respostas: Sim, Não)



proprietario(sergio, livro)

Objetos e Relacionamentos

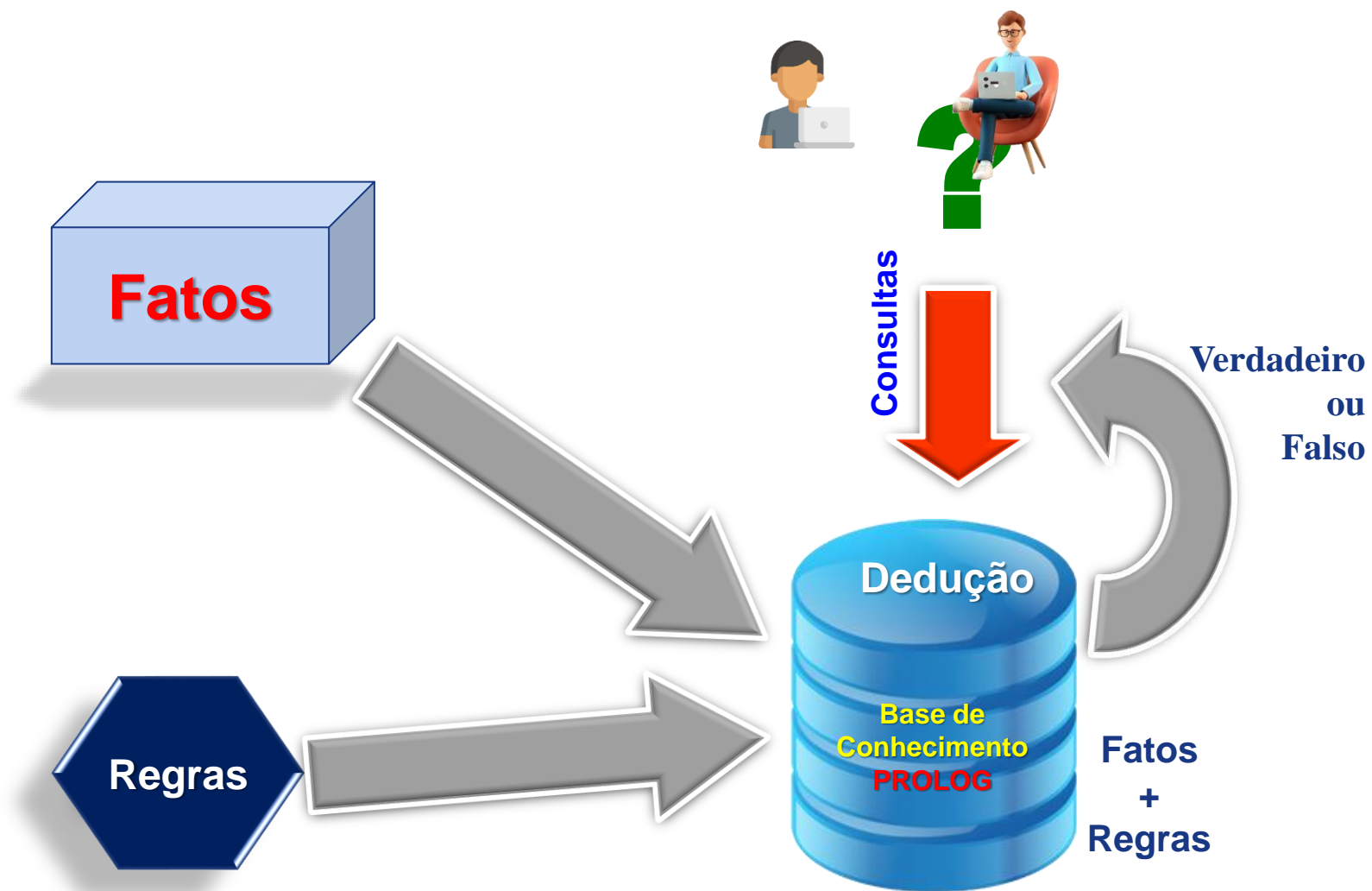
Árvore genealógica



Objetos e Relacionamentos

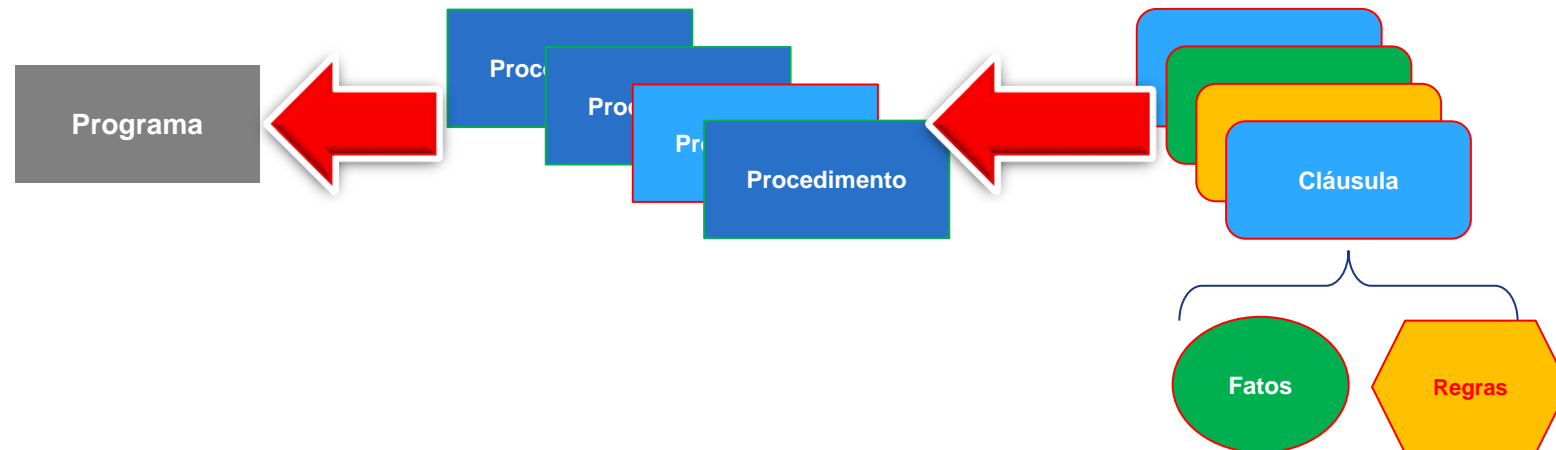


Paradigma Prolog



Programa Lógico

- ❖ Um programa é um conjunto de regras, fatos e deduções
 - regras e fatos para representar informação
 - dedução para responder consultas
- ❖ Programas
 - Programas consistem em um conjunto de procedimentos
 - Procedimentos consistem em um conjunto de cláusulas
 - Cláusulas é um fato ou uma regra



Programar - Executar

The screenshot shows the SWI-Prolog-Editor window with the file '00-calculadora.pl' open. The editor contains a Prolog program with several arithmetic rules. The execution console at the bottom shows the results of queries, including an error for an unknown procedure and successful results for the 'soma' (sum) and 'quadrado' (square) rules.

```
1 % Arquivo:      calculadora.pl
2 % Prof. Ausberto S. Castro Vera
3 % Disciplina : Paradigmas de Ling. de Programacao
4 % UENF-CCT-LCMAT-CC
5 % Date: 2023
6 % Aluno: Fulano da Silva
7 %-----
8
9 soma(A,B,Resultado):- Resultado is A+B.
10 resta(X,Y,Resultado) :- Resultado is X-Y.
11 multip(C,D,Resultado):- Resultado is C*D.
12 quadrado(X,Resultado):- Resultado is X*X.
13 divide(X,Y,Resultado):- Resultado is X/Y, Y > 0.
14
15
```

Linha: 1 Coluna: 1 Inserir ANSI/Windows

ERROR: Unknown procedure: soma/3 (DWIM could not correct goal)
?- consult('00-calculadora').

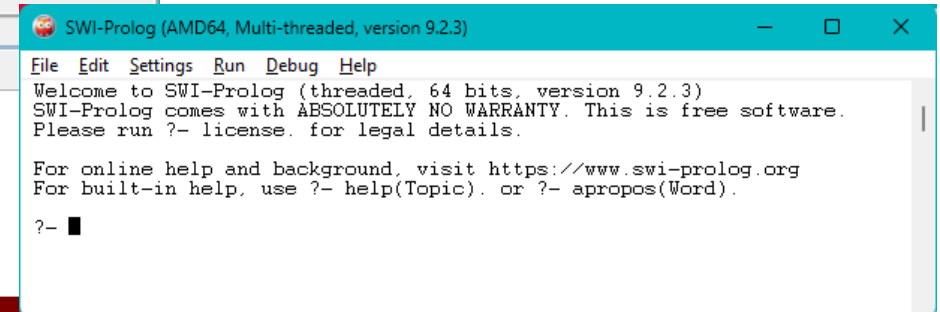
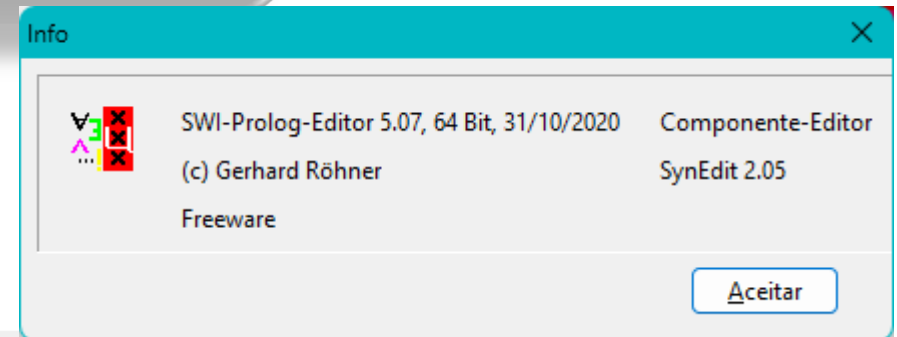
?- soma(3,4,Resultado).
Resultado = 7 ;
false.

?- resta(8,3,Dif).
Dif = 5 ;
false.

?- quadrado(9,Q).
Q = 81 ;
false.

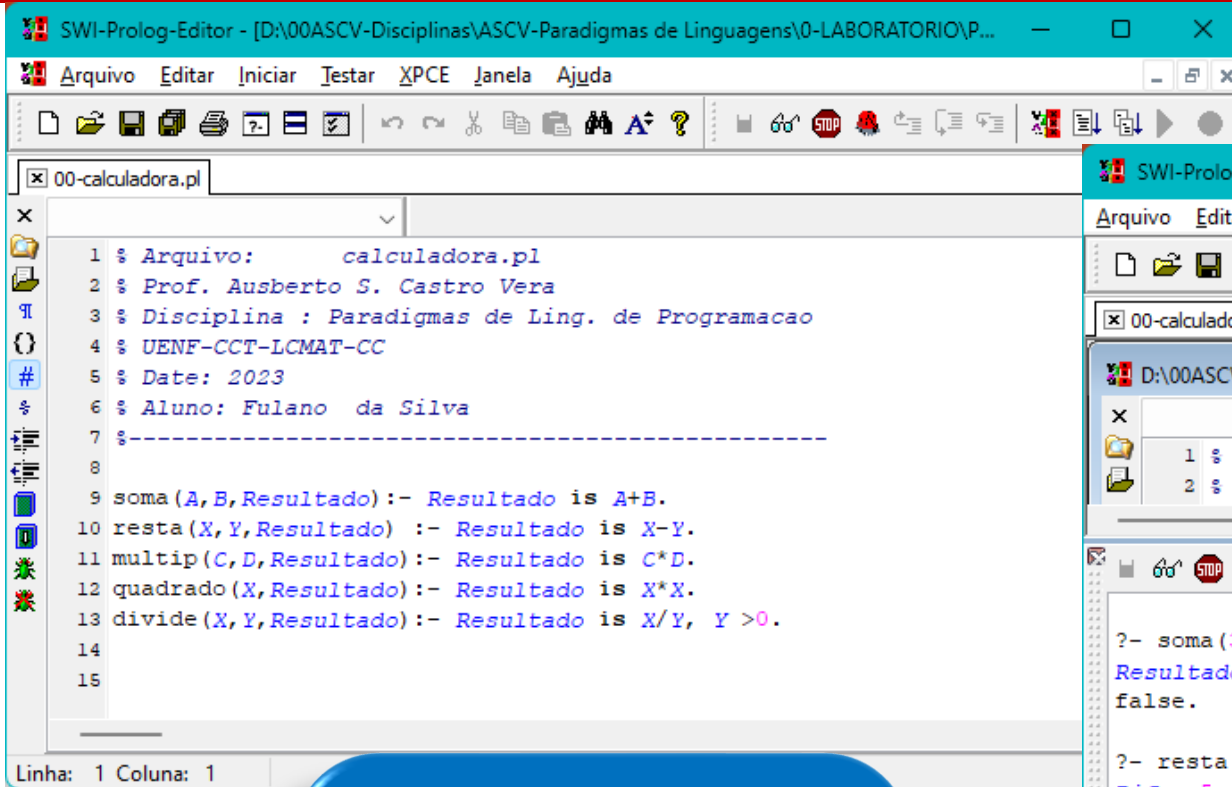
?- |

Programar significa
declarar axiomas e regras



Executar um programa
significa fazer consultas

Programar - Executar



SWI-Prolog-Editor - [D:\00ASCV-Disciplinas\ASCV-Paradigmas de Linguagens\0-LABORATORIO\Prolog\00-calculadora.pl]

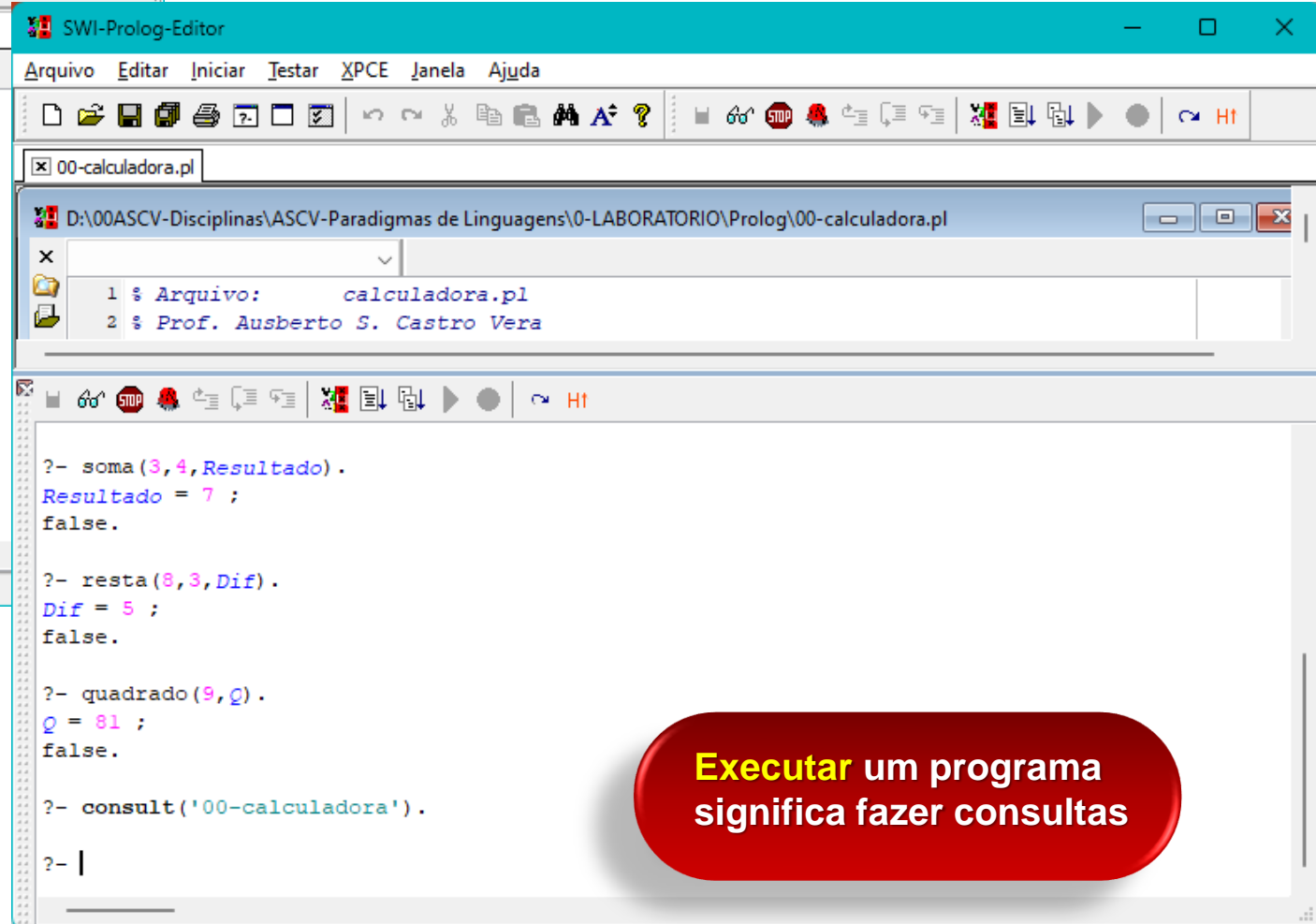
Arquivo Editar Iniciar Testar XPCE Janela Ajuda

00-calculadora.pl

```
1 % Arquivo:      calculadora.pl
2 % Prof. Ausberto S. Castro Vera
3 % Disciplina : Paradigmas de Ling. de Programacao
4 % UENF-CCT-LCMAT-CC
5 % Date: 2023
6 % Aluno: Fulano da Silva
7 %-----
8
9 soma(A,B,Resultado):- Resultado is A+B.
10 resta(X,Y,Resultado) :- Resultado is X-Y.
11 multip(C,D,Resultado):- Resultado is C*D.
12 quadrado(X,Resultado):- Resultado is X*X.
13 divide(X,Y,Resultado):- Resultado is X/Y, Y > 0.
14
15
```

Linha: 1 Coluna: 1

Programar significa
declarar **axiomas e regras**



SWI-Prolog-Editor

Arquivo Editar Iniciar Testar XPCE Janela Ajuda

00-calculadora.pl

D:\00ASCV-Disciplinas\ASCV-Paradigmas de Linguagens\0-LABORATORIO\Prolog\00-calculadora.pl

```
1 % Arquivo:      calculadora.pl
2 % Prof. Ausberto S. Castro Vera
```

?- soma(3,4,Resultado).
Resultado = 7 ;
false.

?- resta(8,3,Dif).
Dif = 5 ;
false.

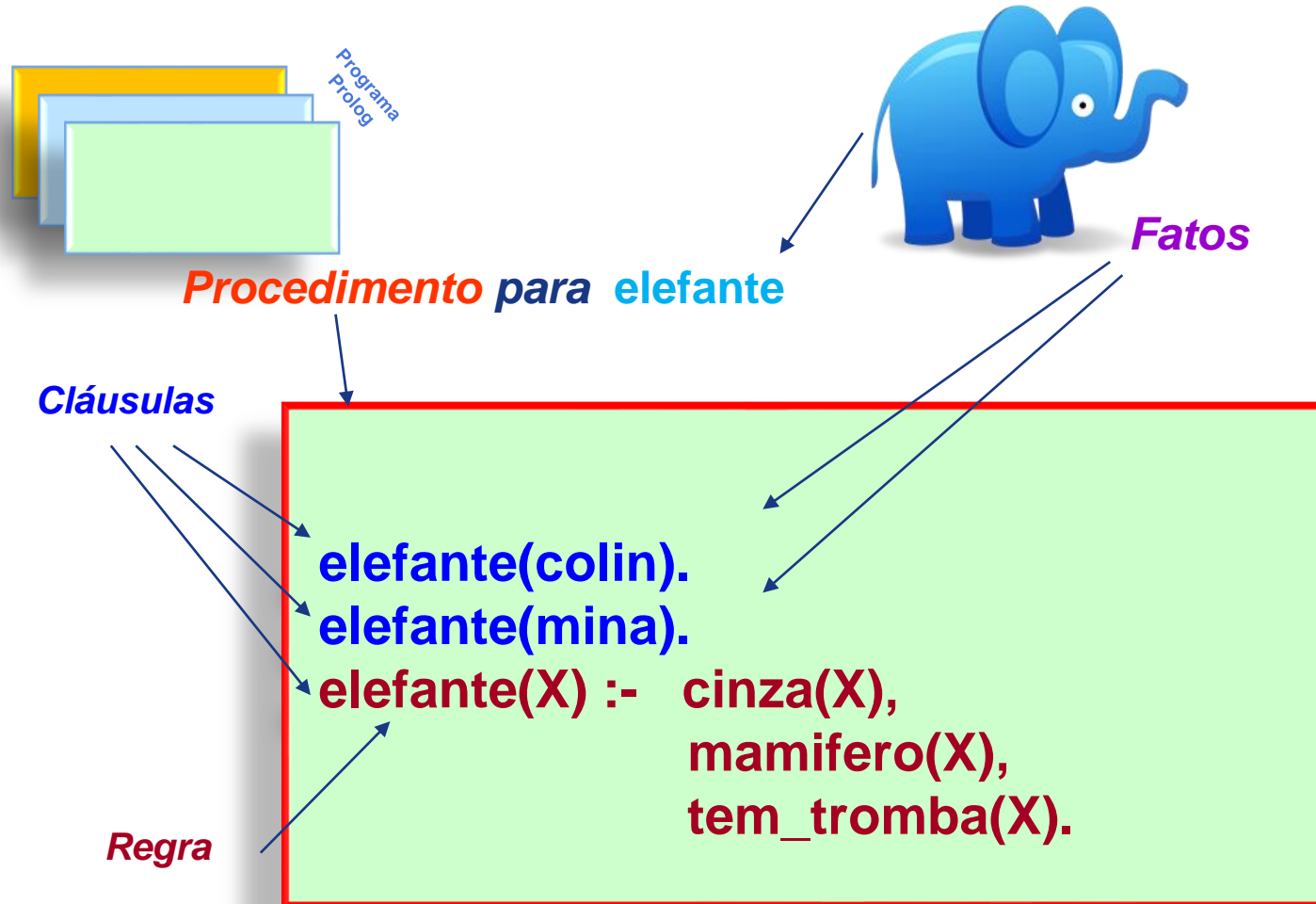
?- quadrado(9,Q).
Q = 81 ;
false.

?- consult('00-calculadora').

?- |

Executar um programa
significa fazer **consultas**

Programa PROLOG



Predicado é a construção utilizada para declarar algo a respeito de objetos

Programa PROLOG

❖ Regras

`avo(A,C) :- pai(A,B), pai(B,C).`

❖ Fatos

`pai(Jorge, João).`

`pai(João, Guilherme).`

❖ Consultas

`?- avo(Jorge, Guilherme)`

`yes`

`?- avo(Guilherme, Jorge)`

`not`



Fatos

❖ Base do conhecimento

- Knowledge Base (KB)
- Coleção de fatos

❖ Fato

- Utilizados para estabelecer propriedades que são incondicionalmente verdadeiras em um domínio de interesse



❖ São conhecidas como Cláusulas de Horn sem cabeça

- **estrela(sol).**
- **ponto(12,8).**
- **funcionario("pedro")**

❖ Representam asserções (afirmações) incondicionais

Fatos – Sintaxe

predic(arg1, arg2, ... argN).

- ❖ **predic**
 - Nome do predicado
- ❖ **arg1, arg2, ..., argN**
 - argumentos
- ❖ **N**
 - Aridade
- ❖ **Predicado de aridade 0**
 - pred.
- ❖ **Termina com ponto!**
- ❖ **Os argumentos podem ser qualquer termo PROLOG válido**
 - **Inteiro**
 - **Átomo**
 - **Texto constante começando com letra minúscula**
 - **Variável**
 - **Começa com letra maiúscula ou underline (_)**
 - **Estrutura**
 - **Termo complexo**

Fatos



separados por vírgulas

gosta_de(**pedro**, **suzana**).

Relacionamento
(predicado)

Objetos
(argumentos)

ponto no final

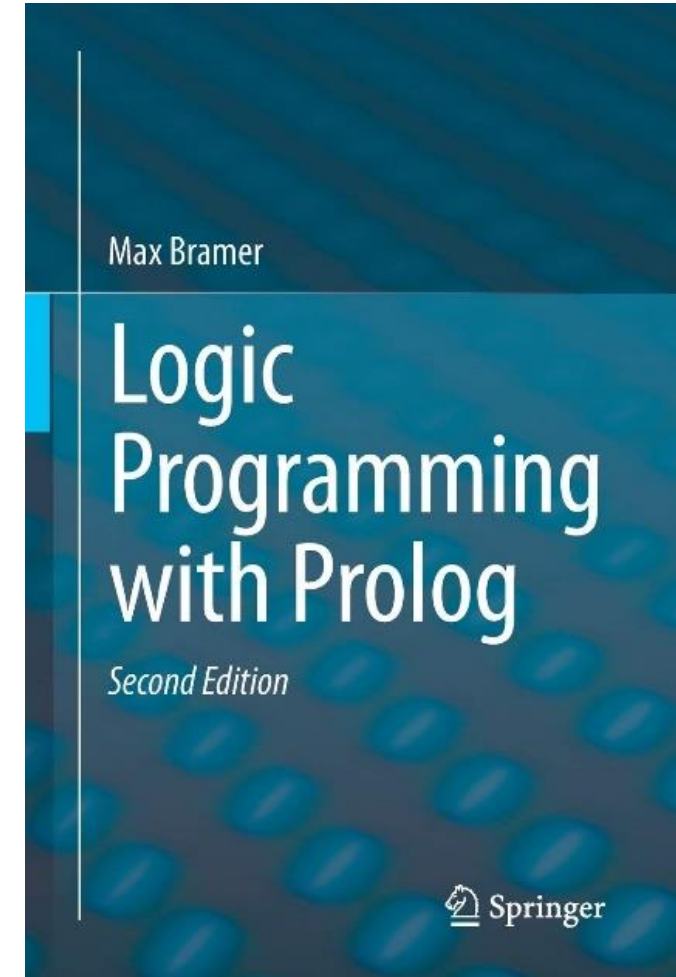
Fatos



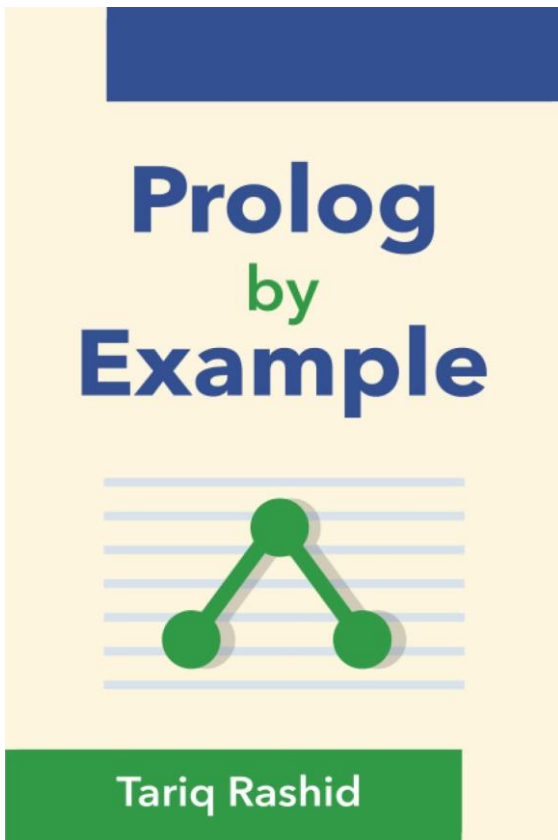
Fatos

```
book('prolog', 'max bramer', 'springer', 2013).  
autor('max', 'bramer').  
publicador('springer').  
publicador('prentice-hall').
```

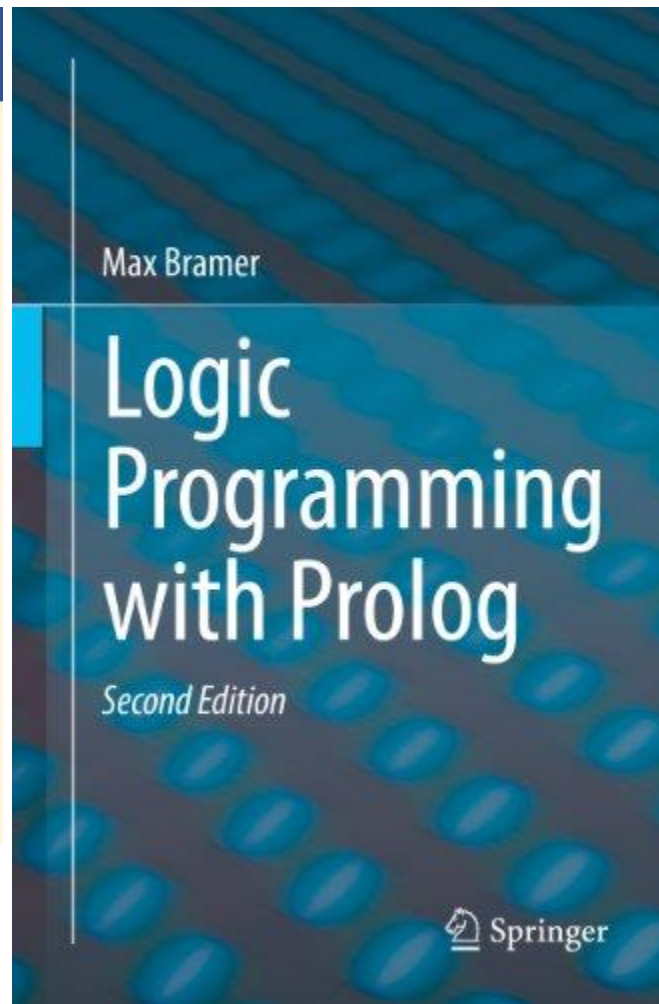
```
book(T,A,publisher(C,rome),Date).  
natural(0).  
inteiro(-4).  
real(3.141516).  
pi(3.141516).  
pai(abraham, isaque).  
filho(isaque, esau).  
verdura(alface).
```



Exercício



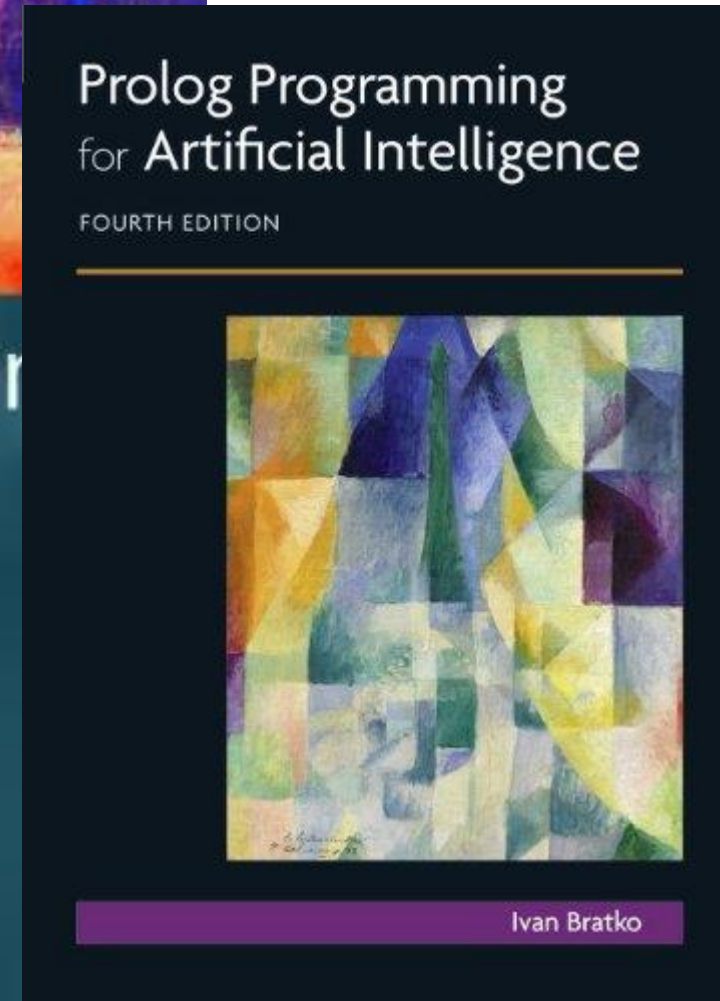
2023



2013



2003



2011

Base de conhecimento PROLOG

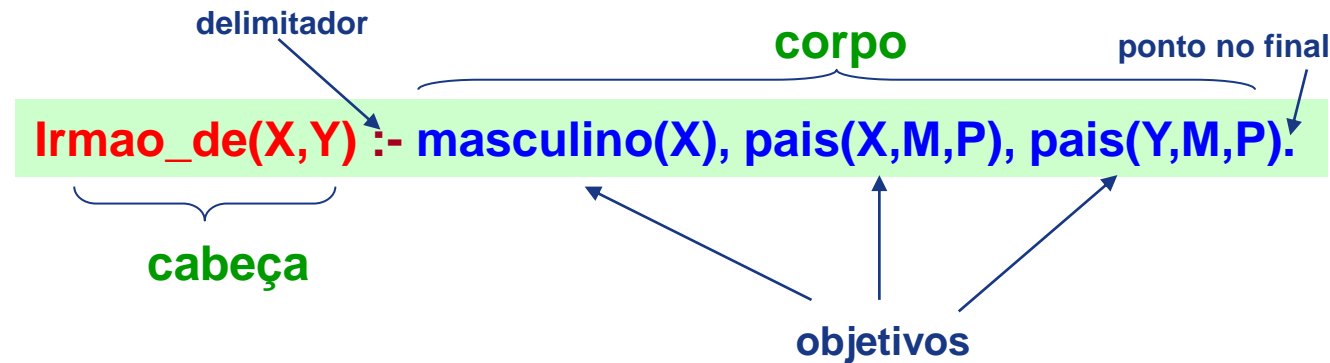
Regras - Sintaxe

cabeça :- corpo

- ❖ **Cabeça**
 - Definição de predicado (similar a um fato)
- ❖ **O símbolo :-**
- ❖ **Corpo**
 - Um ou mais objetivos (goals)

Regras

- ❖ Descrevem relacionamentos, utilizando outros relacionamentos
- ❖ Estabelecem dependência: *..A..., se ...B..*



X é irmão de Y se X é masculino X tem como mãe M e pai P
Y tem como mãe M e pai P

Regras

- ❖ Conhecidas como *Cláusulas de Horn com cabeça*
- ❖ Utilizam variáveis para generalizar seu significado
- ❖ São asserções condicionais
 - A cabeça é verdadeira se o corpo for verdadeiro
- ❖ Definem como novos fatos podem ser criados
- ❖ As vírgulas (,) no corpo representam *conjunções*
 - Exemplo:

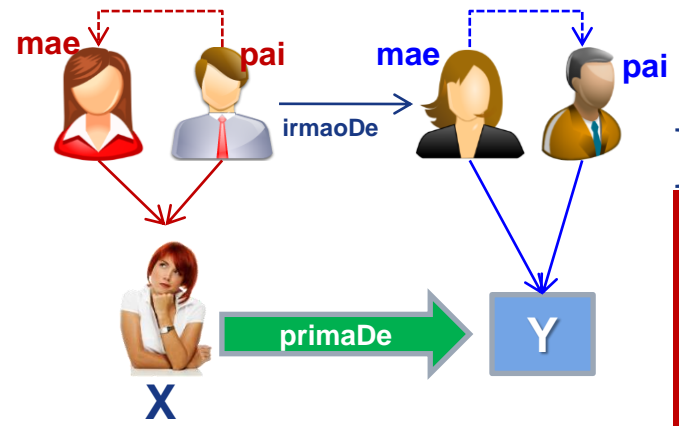
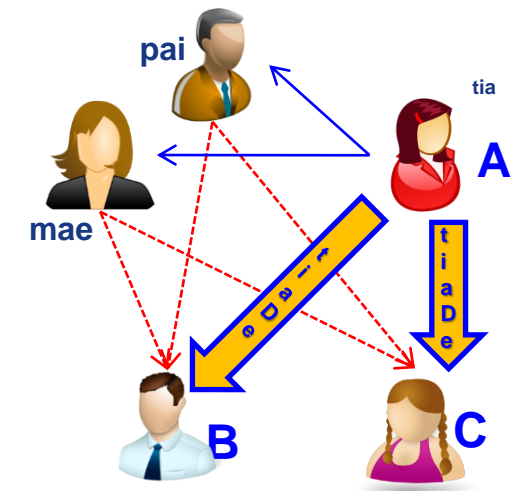
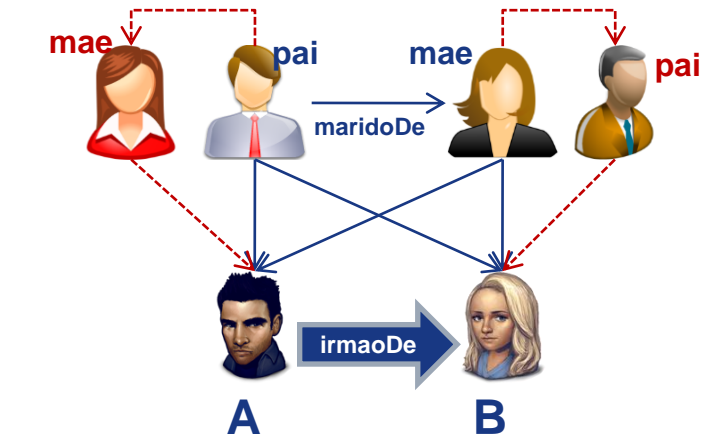
mae(M,X), mae(M,Y), pai(P,X), pai(P,Y).

significa

mae(M,X) **AND** mae(M,Y) **AND** pai(P,X) **AND** pai(P,Y).

Regras

```
avo(A,C) :- pai(A,B) , pai(B,C) .
irmao(X,Y) :- mae(M,X) , mae(M,Y) , pai(P,X) ,
pai(P,Y) .
filha(F,P) :- pai(P,F) , mulher(F) , homem(P) .
filho(F,P) :- pai(P,F) , homem(F) , homem(P) .
elefante(X) :- cinza(X) , mamifero(X) ,
temTromba(X) .
natural_number(N) :- integer(N) , N >= 0.
                                //factorial(0,1) .
factorial(A,B) :- A > 0,  C is A-1,
factorial(C,D) , B is A*D.
max(A,B,M) :- A < B, M = B.
max(A,B,M) :- A >= B, M = A.
```



FAMILIAS

Regras:
irmao(A,B) :- ...
prima(X,Y) :- ...
tia(A,B):- ...

Consultas

- ❖ Significa perguntar à base de dados do sistema sobre determinadas informações
- ❖ Ocorrem em forma interativa
- ❖ Realizadas depois de carregar o programa (fatos e regras)
- ❖ prompt **?-**



```
SWI-Prolog Editor - [E:\00ASCV-Disciplinas\ASCV-Paradigmas de Linguagens\0-LABORATORIO\Prolog\00-calculadora.pl]
Arquivo  Editar  Iniciar  Testar  XPCE  Janela  Ajuda
X 00-calculadora.pl
1 % Arquivo:    calculadora.pl
2 % Prof. Ausberto S. Castro Vera
3 % Disciplina : Paradigmas de Ling. de Programacao
4 % UENF-CCT-LCMAT-CC
5 % Date: Setembro 2022
6 % Aluno: Fulano da Silva
7 %-----
8
9 soma(A,B,Resultado):- Resultado is A+B.
10 resta(X,Y,Resultado) :- Resultado is X-Y.
11 multip(C,D,Resultado):- Resultado is C*D.
12 quadrado(X,Resultado):- Resultado is X*X.
13 divide(X,Y,Resultado):- Resultado is X/Y, Y > 0.
14
15
Linha: 15  Coluna: 72  Inserir  ANSI/Windows
?- soma(2,7,ResultadoSoma).
ResultadoSoma = 9 ;
false.

?- resta(18,5,ResultadoResta).
ResultadoResta = 13 ;
false.

?- quadrado(8,ResultadoQuadr).
ResultadoQuadr = 64 ;
false.

?- |
```

Base de Dados

- ❖ É o conjunto de fatos no sistema PROLOG
- ❖ Contém os fatos a partir dos quais, as consultas serão respondidas
 - O sistema PROLOG constrói seu conhecimento a partir destes fatos
 - O programador PROLOG é responsável pela sua exatidão



```
humano(pele) .  
brasileiro(pele) .  
jogador(pele) .  
presidente(lula, brasil) .  
cantor(romario) .  
campeao(barcelona, espanha) .  
Campeao(internacional, brasil) .
```

```
mae(Teresa) .  
dispositivo(papel)  
rede(caneta)  
pai(Pedro) .  
mamifero(tigre) .  
profeta(Jeremias) .  
igreja(Laodiceia, Grecia) .
```

Programa Exemplo: casa.pl

```
% Author: Ausberto S. Castro Vera
% UENF, Ciência da Computação, 05/2019
```

```
%----- FATOS -----
```

```
lugar(sala).
lugar(cozinha).
lugar(escritorio).
lugar(corredor).
lugar('sala de jantar').
lugar(celeiro).
lugar(jardim).
```

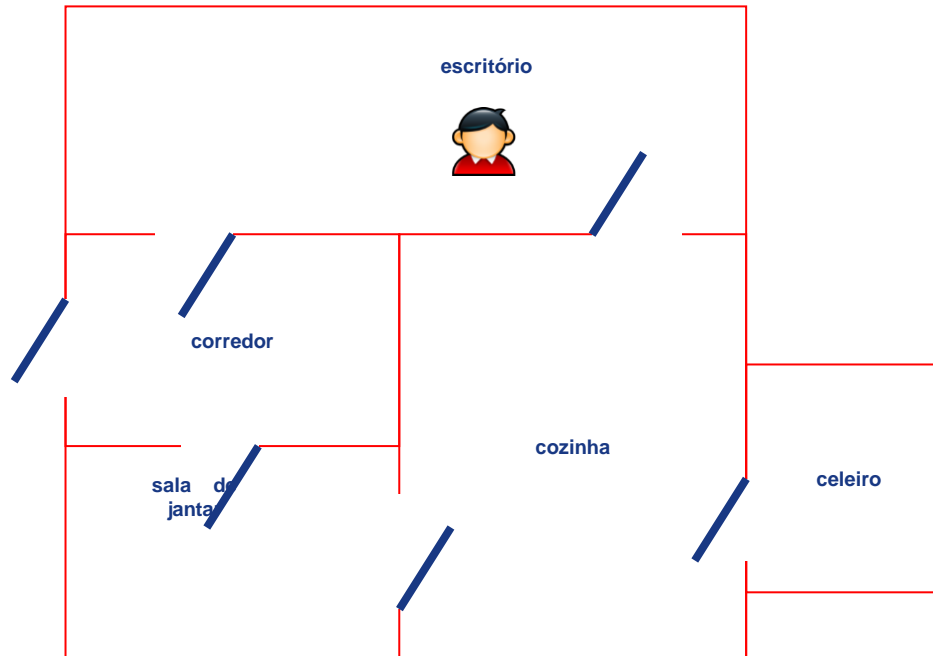
```
porta(escritorio, corredor).
porta(cozinha, escritorio).
porta(corredor, 'sala de jantar').
porta(cozinha, celeiro).
porta('sala de jantar', cozinha).
```

```
localizacao(escrivania, escritorio).
localizacao(laranja, cozinha).
localizacao(lanterna, escritorio).
localizacao('maq de lavar', celeiro).
localizacao(sabao, 'maq de lavar').
localizacao(verdura, cozinha).
localizacao(biscoitos, cozinha).
localizacao(computador, escritorio).
```

```
comestivel(laranja).
comestivel(biscoitos).
```

```
gosto_amargo(verdura).
```

```
aqui(cozinha).
```



```
%%----- REGRAS -----
```

```
conectar(X,Y) :- porta(X,Y).
```

```
conectar(X,Y) :- porta(Y,X).
```

```
lista_coisas(Lugar) :-
```

```
    localizacao(X, Lugar),
    tab(2),
    write(X),
    nl,
    fail.
lista_coisas(Qualquerlugar).
```

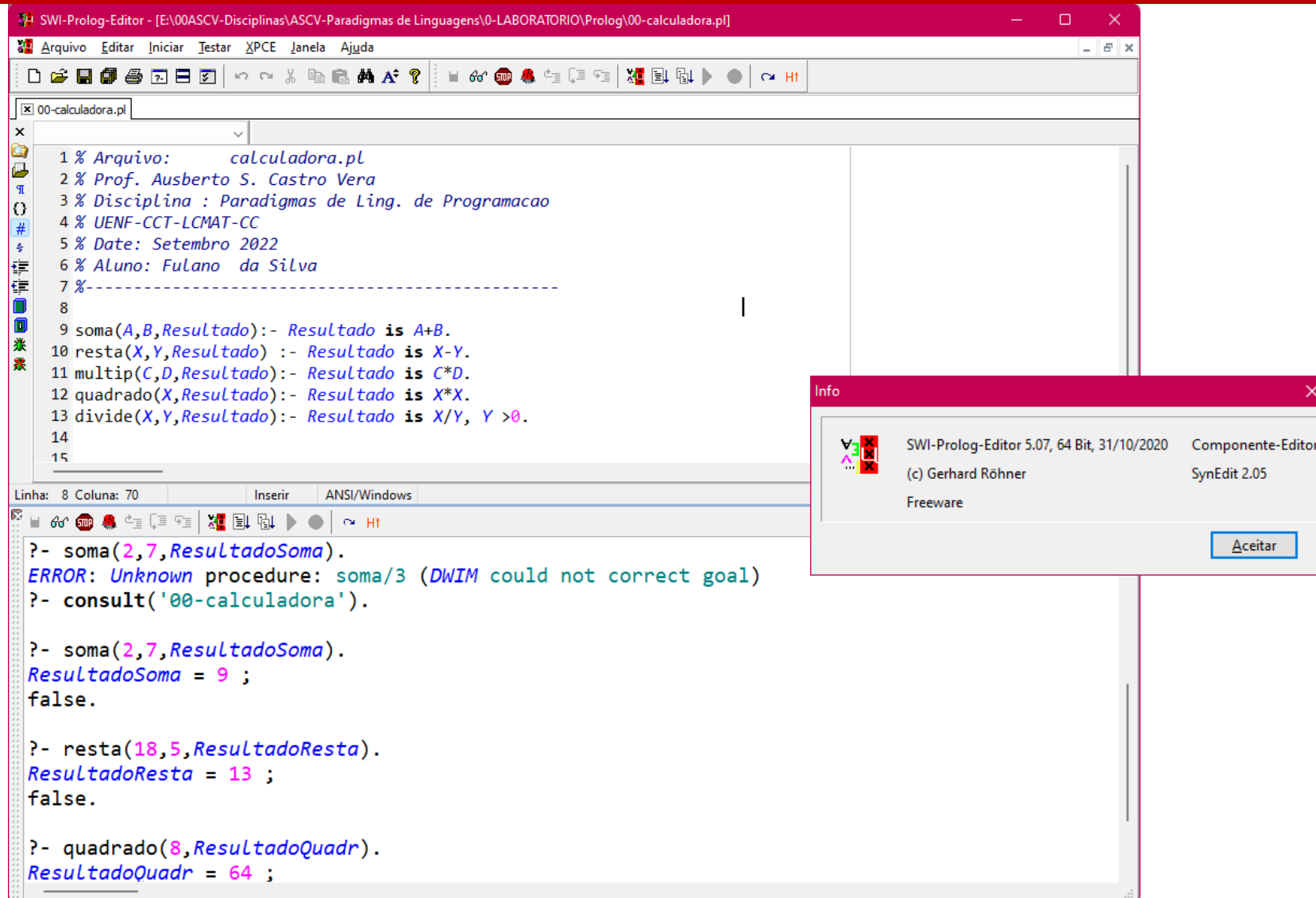
```
lista_portas(Lugar) :-
```

```
    conectar(Lugar, X),
    tab(2),
    write(X),
    nl,
    fail.
lista_portas(_).
```

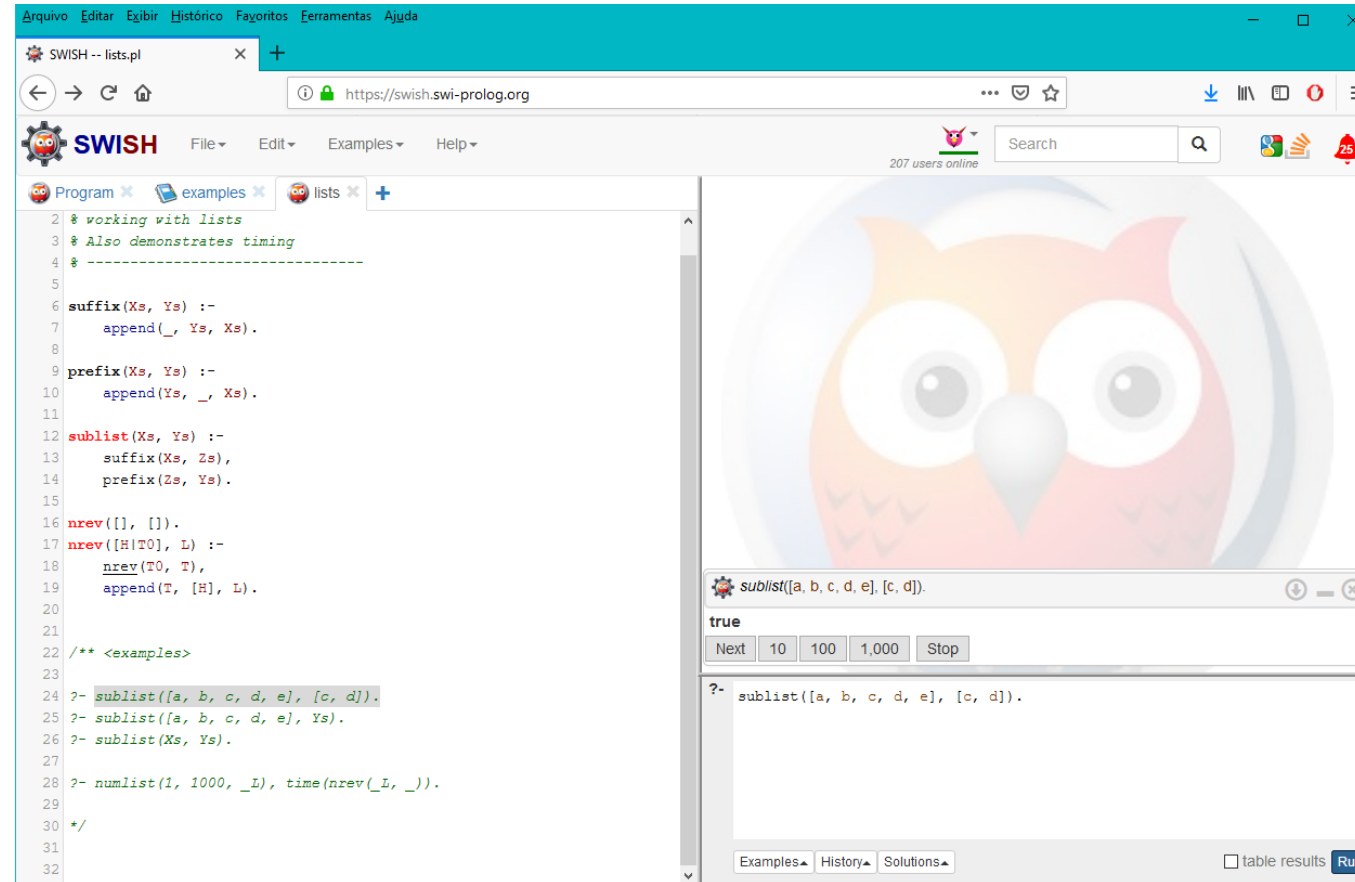
```
observar :- aqui(Lugar),
```

```
    write('Voce esta na '), write(Lugar),
    nl,
    write('Voce pode ver:'), nl,
    lista_coisas(Lugar),
    write('Voce pode ir para:'), nl,
    lista_portas(Lugar).
```

SWI Prolog 9.0.4 + Editor 5.07 (64 bits)



SWISH - <https://swish.swi-prolog.org/>



The screenshot displays the SWISH web interface in a browser window. The address bar shows the URL <https://swish.swi-prolog.org>. The interface includes a menu bar with options like Arquivo, Editar, Exibir, Histórico, Favoritos, Ferramentas, and Ajuda. Below the menu, there's a search bar and a notification for 207 users online. The main area is divided into two panes. The left pane shows a Prolog program with the following code:

```
2  * working with lists
3  * Also demonstrates timing
4  * -----
5
6  suffix(Xs, Ys) :-
7      append(_, Ys, Xs).
8
9  prefix(Xs, Ys) :-
10     append(Ys, _, Xs).
11
12  sublist(Xs, Ys) :-
13     suffix(Xs, Zs),
14     prefix(Zs, Ys).
15
16  nrev([], []).
17  nrev([H|T0], L) :-
18     nrev(T0, T),
19     append(T, [H], L).
20
21
22 /** <examples>
23
24 ?- sublist([a, b, c, d, e], [c, d]).
25 ?- sublist([a, b, c, d, e], Ys).
26 ?- sublist(Xs, Ys).
27
28 ?- numlist(1, 1000, _L), time(nrev(_L, _)).
29
30 */
31
32
```

The right pane shows the execution results for the query `sublist([a, b, c, d, e], [c, d]).`. The result is `true`. Below the result, there are buttons for `Next`, `10`, `100`, `1,000`, and `Stop`. At the bottom of the right pane, there are tabs for `Examples`, `History`, and `Solutions`, along with a checkbox for `table results` and a `Run!` button.

Sintaxe Prolog

- ❖ Um **fato** é um termo sem variáveis

`gato.`

`verde(papagaio).`

`habitat(tigre,mato).`

- ❖ **Regras** são definidas por uma cabeça seguida do símbolo **`:-`** seguida da *conjunção* de termos, os quais são as condições para que a cabeça seja verdadeira

`pai(A,B) :- parente(A,B), homem(A).`

`natural(N) :- inteiro(N), N > 0.`

- ❖ **Consulta** é a *conjunção* de termos

`?- habitat(gato, floresta).`

Sintaxe Prolog

<fato> ::= <termo>.

<regra> ::= <termo> :- <termos>.

<consulta> ::= <termos>.

<termos> ::= <termo>. | <termo> , <termos>

<termo> ::= <numero>

| <atomo> | <variavel>

| <atomo> (<termos>)

<predicado> ::= atomo

<atomo> ::= nomepred | nomepred(t1, t2, ..,tn)

<variavel> ::= identificador começando com letra minúscula

Compiladores-Interpretadores

- ❖ **SWI Prolog**
 - <http://www.swi-prolog.org/>
- ❖ **Amzi!**
 - <http://www.amzi.com/>
- ❖ **BProlog**
 - <http://www.sci.brooklyn.cuny.edu/~zhou/bprolog.html>
- ❖ **GNU Prolog**
 - <http://pauillac.inria.fr/~diaz/gnu-prolog/>
- ❖ **Poplog**
 - <http://www.poplog.org/>
- ❖ **Strawberry Prolog**
 - <http://www.dobrev.com/>
- ❖ **Visual Prolog**
 - <http://www.visual-prolog.com/>



Prof. Dr. Ausberto S. Castro Vera
Ciência da Computação
UENF-CCT-LCMAT
Campos, RJ

ascv@uenf.br
ausberto.castro@gmail.com

