

Prolog



Equipe:

Bruno Zandona

Gabriel Vargas

Nilson Deon

Saulo de Moura



PUC Minas

Abril / 2023

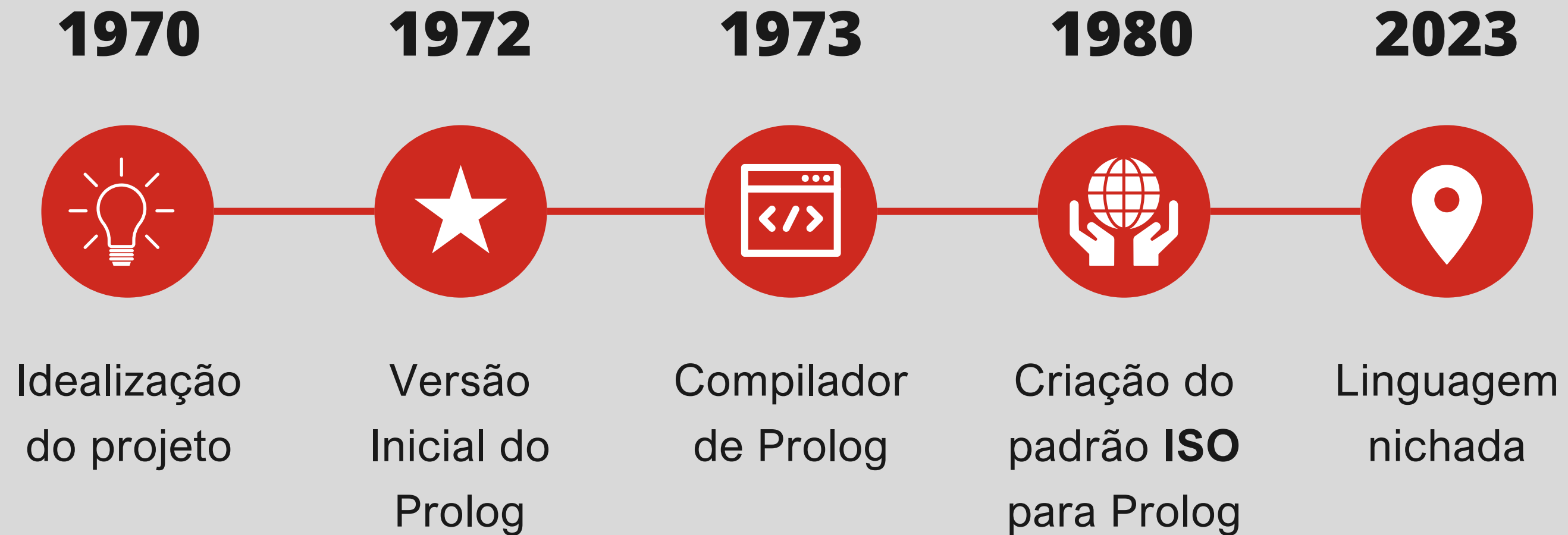
Sumário

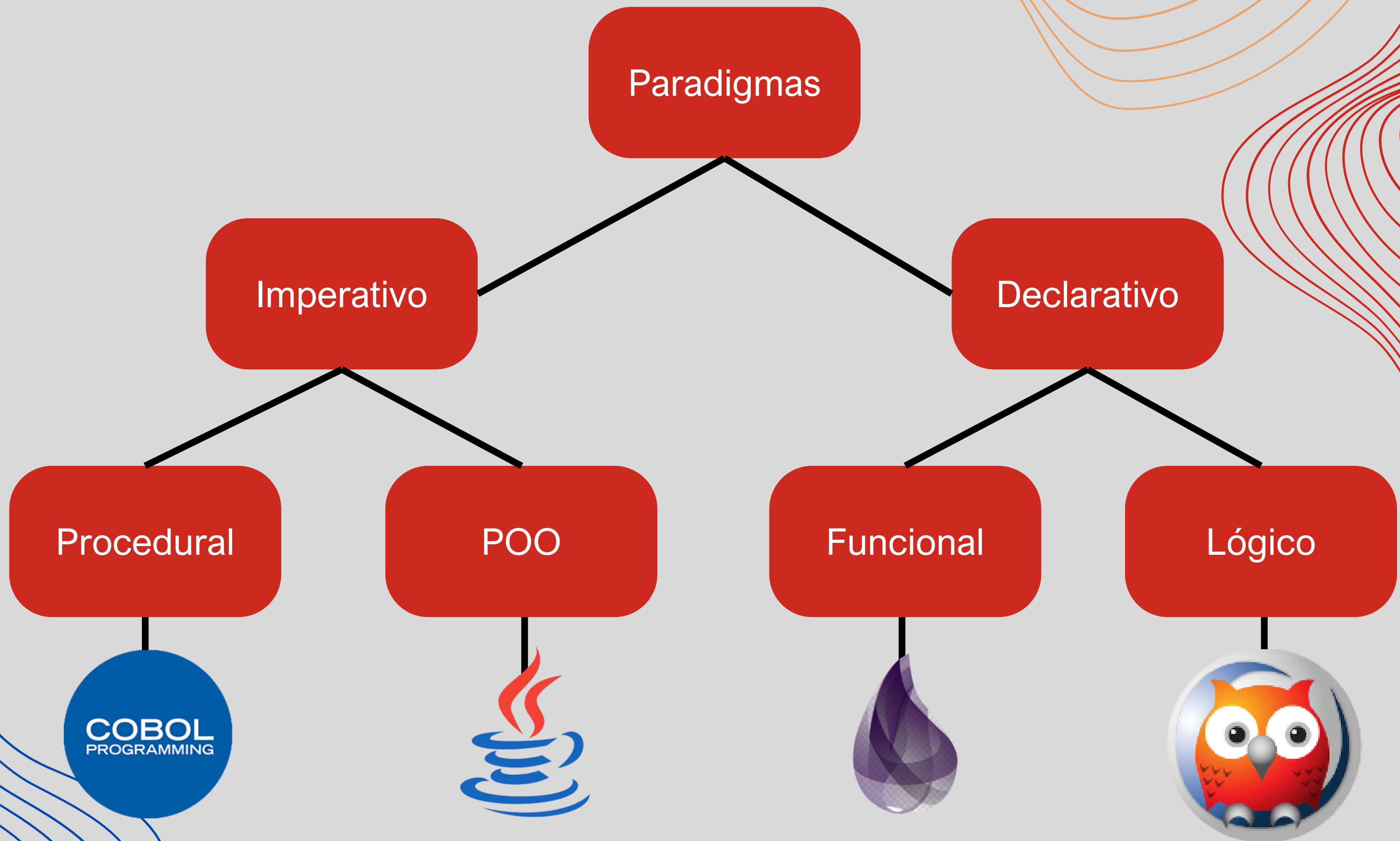
- História e cronologia
- Paradigmas
- Linguagens relacionadas
- Principais utilizações
- Projetos em Prolog
- Elementos base
- Tipos de dados
- Operadores
- Prolog x SQL
- Alguns predicados
- Backtracking
- Recursividade
- Estrutura condicional e de repetição
- Instalando a IDE
- Exemplos práticos
- Trabalho futuro
- Considerações finais
- Bibliografia

História

- Idealização do projeto em 1970
- Lançado em 1972 (mesmo ano de C)
- Q-Systems
- Processamento de linguagens naturais
- Universidade de Marselha
- Criadores:
 - Alain Colmerauer
 - Robert Pasero
 - Phillipe Roussel
 - Jean Trudel
 - Robert Kowalski
- **PRO**grammation en **LOG**ique

Cronologia





Paradigmas

- Declarativo
 - Instrui "o que fazer" e não "como fazer"
 - Legível
 - Redução da mutabilidade
- Lógico
 - Declara um conjunto de fatos a serem provados por teoremas
 - Problemas complexos de maneira natural e concisa
 - Capacidade de realizar inferências lógicas
 - Capacidade de lidar com informações incompletas ou incertas

Linguagens Relacionadas

- Planner
 - Primeira linguagem lógica
- Q-Systems
 - Linguagem criada pelo Colmerauer
- Lisp
 - Introduziu a manipulação de dados simbólicos com base em operações lógicas.
- Mercury
 - Influenciado por Prolog e Haskell
- Oz
 - Linguagem multiparadigma também influenciada por Prolog

Principais Utilizações

- Inteligência Artificial
- Processamento de Linguagem Natural
- Sistemas Especialistas
- Automação de Projetos
- Bioinformática
- Banco de Dados

Projetos em Prolog

- IDP (Integrating Deductive and Probabilistic Reasoning)
 - Sistema de **raciocínio automatizado**
 - Raciocínio **dedutivo** e o raciocínio **probabilístico** para resolver problemas
- ACE (Automated Communication Engine)
 - Sistema de **processamento de linguagem natural**
 - Permite interagir com o computador de maneira **natural e eficiente**
- Mycin
 - Sistema **especialista**
 - Diagnosticar doenças infecciosas e recomendar tratamentos
- SHOP (Simple Hierarchical Ordered Planner)
 - Sistema de **planejamento automatizado**
 - Pioneiro em técnicas de planejamento automatizado hoje em dia

Projetos em Prolog

- Prolog Chess
 - **Jogo de xadrez** que demonstra as capacidades da programação lógica.



Diponível em: <<http://www.fraber.de/>>. Acesso em: 03 abr. 2023.

Elementos base

% Fatos:

```
mulher(maria).  
homem(joão).  
homem(pedro).  
pai(joão, pedro).  
mãe(maria, pedro).
```

% Regras:

```
filho(X, Y) :- mãe(Y, X); pai(Y, X).
```

/*

Consultas:

? - pai(X, pedro).

X = joão.

*/

Elementos base

% Fatos

mulher(maria).

homem(joão).

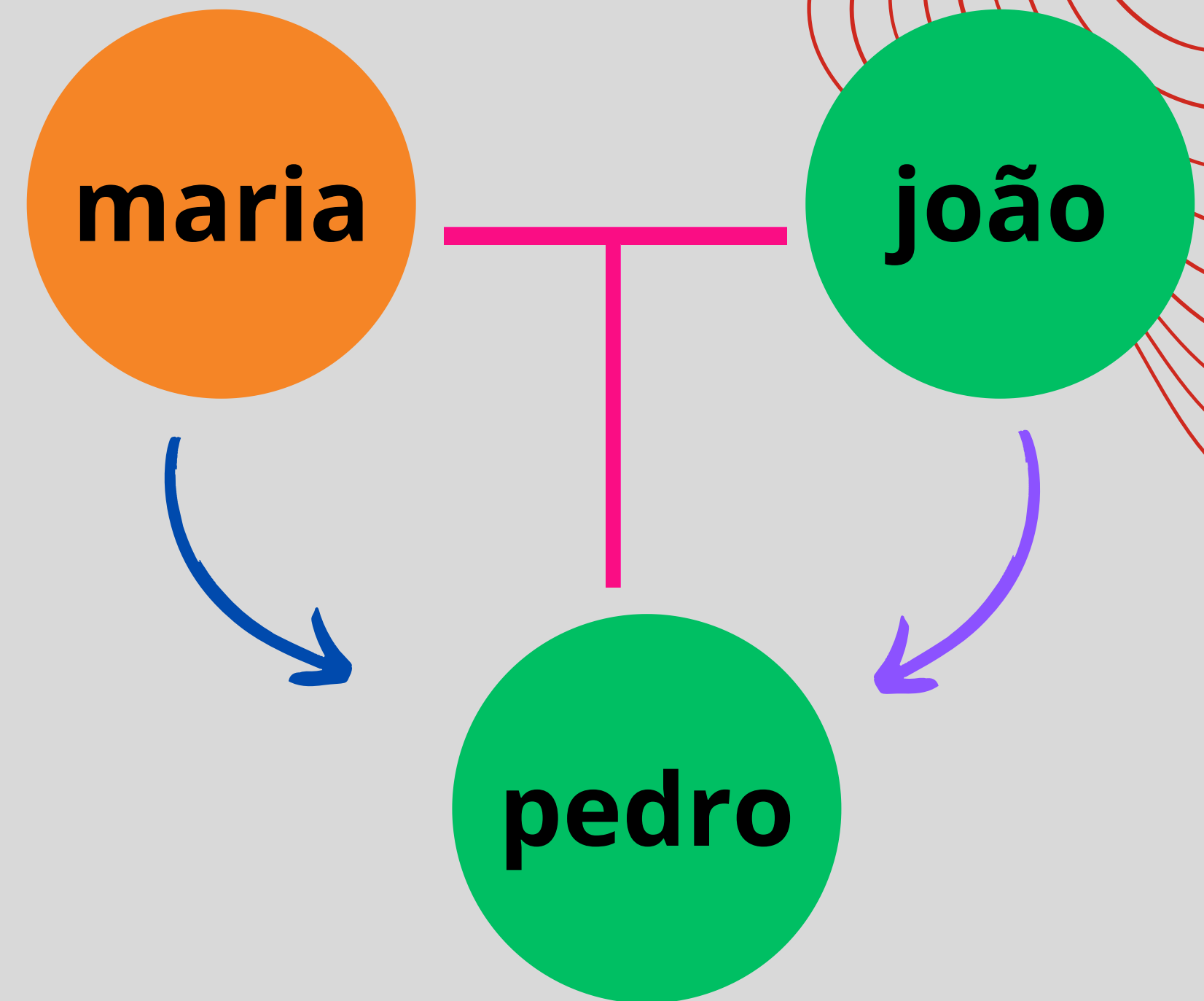
homem(pedro).

pai(joão, pedro).

mãe(maria, pedro).

% Regras

filho(X, Y) :- mãe(Y, X); pai(Y, X).



Elementos base

```
1  % Fatos
2  mulher(maria).
3  homem(joão).
4  homem(pedro).
5  pai(joão, pedro).
6  mãe(maria, pedro).
7
8  % Regras
9  filho(X, Y) :- mãe(Y, X); pai(Y, X).
10
```

exemplo01.pl

Tipos de dados

Átomos:

- Sequência de caracteres
- Começam com letra minúscula
- String se entre aspas duplas
- `maria`, `joão`, `'Pedro'`, `"String"`, `<-->`, `:`, `quadrado`, `a1`, `á`

Números:

- Inteiros e Pontos flutuantes
- `1`, `-2.48`, `0`, `6.5`, `-9`

Tipos de dados

Variáveis:

- Incógnita
- Começa com letra maiúscula
- Não é vinculada a um tipo
- X, Y, NomeProcurado, Lista, Resp

Variável Anônima:

- Valor é irrelevante
- Underline " _ "

Tipos de dados

Lista:

- Cabeça e cauda
- | quantidade indefinida
- Cabeça nunca pode ser vazia, mas cauda pode
- Pode conter tipos diferentes

[]	Vazia
[X]	Único item
[X Y]	Pelo menos um item
[X,Y]	Exatamente dois itens
[X,Y Z]	Pelo menos dois itens

[]
[maria]
[maria, ...]
[maria, 'joão']
[maria, 'joão', ...]

Tipos de dados

Estruturas:

```
8  % Exemplo estruturas
9  diasSemana(seg, ter, qua, qui, sex, sab, dom).
10 aniversario(data(17, março, 2003), nome('Maria da Silva')).
```

```
?- aniversario(Data, nome('Maria da Silva')).
Data = data(17, março, 2003).
```

```
?- aniversario(data(17, março, 2003), Nome).
Nome = nome('Maria da Silva').
```

```
?- aniversario(data(17, Mes, 2003), nome('Maria da Silva')).
Mes = março.
```

IDE

Terminal

Operadores

Aritméticos:

- +** Adição
- Subtração
- *** Multiplicação
- //** Divisão inteira
- /** Divisão real
- mod** Resto divisão inteira
- **** Potenciação
- abs, sin, cos, tan, exp, ln, log, sqrt**

Relacionais:

- @** { **>** Maior que
- <** Menor que
- >=** Maior ou igual que
- =<** Igual ou menor que

Variável < Número < Átomo < String < Composto

Operadores

Relacionais:

$:=$ Igualdade

is Unificação com expressão aritmética (lado direito)

$=$ Unificação

$==$ Idênticos

\neq Diferença

\neq Não idênticos

Operadores

Lógicos:

\+ Not
, And
; Or

```
1  % pessoa(Nome, Idade, Sexo) .  
2  pessoa(ana, 20, f) .  
3  pessoa(eva, 15, f) .  
4  pessoa(bia, 30, f) .  
5  pessoa(leo, 30, m) .  
6  pessoa(ilo, 35, m) .  
7  pessoa(aly, 46, m) .
```

```
?- pessoa(Nome, Idade, Sexo), Idade =< 30, Sexo @=< f, \+ Nome @=< ana.  
Nome = eva,  
Idade = 15,  
Sexo = f ;  
Nome = bia,  
Idade = 30,  
Sexo = f .
```

Prolog x SQL

```
1  % pessoa(Nome, Idade, Sexo).
2  pessoa(ana, 20, f).
3  pessoa(eva, 15, f).
4  pessoa(bia, 30, f).
5  pessoa(leo, 30, m).
6  pessoa(ilo, 35, m).
7  pessoa(aly, 46, m).
```

Liste o nome de todas as mulheres maiores de 18 anos:

```
13  % ?- pessoa(Nome, f, Idade), Idade > 18.
14
15  adultas(Adultas) :-
16      findall(Nome, (pessoa(Nome, f, Idade), Idade > 18), Tmp),
17      sort(Tmp, Adultas).
```

```
1  SELECT nome
2  FROM pessoas
3  WHERE sexo = 'f' AND idade > 18
4  ORDER BY nome
5
```

```
?- adultas(Adultas).
Adultas = [ana, bia, eva].
```


Prolog x SQL

```
1  % Base de conhecimento dinâmica:
2
3  % Fatos
4  joga(pelé,futebol).
5  joga(guga,tênis).
6  joga(tiago,vôlei).
7
8  % Regras
9  :- dynamic joga/2.
10 esporte(X) :- joga(_,X).
```

- asserta(<fato>).
- assertz(<fato>).
- retract(<fato>).

Alguns predicados

IO:

- read(X).
- write(X).
- nl().

Lista:

- length(Tamanho, <lista>).
- append(<lista1>, <lista2>, <listaFinal>).
- reverse(<lista>, <listaReversa>).
- sort(<lista>, <listaOrdenada>).

Backtracking

```
1  % Fatos
2  d(0).
3  d(1).
4
5  % Listar números binários de 3 dígitos
6  binário([A,B,C]) :- d(A), d(B), d(C).
7
8  % Listar números binários de 3 dígitos: [0,3]
9  binário0_3([A,B,C]) :- d(A), !, d(B), d(C).
```

```
?- binário(NumBinario).
NumBinario = [0, 0, 0] ;
NumBinario = [0, 0, 1] ;
NumBinario = [0, 1, 0] ;
NumBinario = [0, 1, 1] ;
NumBinario = [1, 0, 0] ;
NumBinario = [1, 0, 1] ;
NumBinario = [1, 1, 0] ;
NumBinario = [1, 1, 1].

?- binário0_3(NumBinario0_3).
NumBinario0_3 = [0, 0, 0] ;
NumBinario0_3 = [0, 0, 1] ;
NumBinario0_3 = [0, 1, 0] ;
NumBinario0_3 = [0, 1, 1].
```

Recursividade

```
6  % Regra para calcular fatorial de um número
7  fat(0, 1) :- !.
8  fat(Numero, Resultado) :-
9      Numero > 0,
10     NewNumero is Numero - 1,
11     fat(NewNumero, Resultado_tmp),
12     Resultado is Numero * Resultado_tmp, !.
13 fat(_, 0) :- write('Valor inválido!').
```

```
?- fat(5, Fat).
Fat = 120.
```

```
?- fat(1, Fat).
Fat = 1.
```

```
?- fat(0, Fat).
Fat = 1.
```

```
?- fat(-3, Fat).
Valor inválido!
Fat = 0.
```

```
?- fat(4.2, Fat).
Valor inválido!
Fat = 0.0.
```

Estrutura Condicional

Existe if/else em Prolog?

```
1  % Regras
2  if(Condition,Then,_) :- Condition, !, Then.
3  if(_,_,Else) :- Else.
4
5  paridade(X)      :- if(X mod 2 == 0, write(par), write(ímpar)).
6  múltiplo(_, 0) :- write(não), !.
7  múltiplo(X, Y) :- if(X mod Y == 0, write(sim), write(não)).
```

```
?- paridade(2).
par
true.
```

```
?- paridade(7).
ímpar
true.
```

```
?- múltiplo(9, 3).
sim
true.
```

```
?- múltiplo(5, 2).
não
true.
```

```
?- múltiplo(7, 5).
não
true.
```

```
?- múltiplo(4, 0).
não
true.
```

Estrutura de Repetição

Existe for em Prolog?

```
12  % Regras
13  % Condição de parada
14  for(Start, End, _) :- Start >= End, !.
15
16  % Executa a ação, decrementa contador
17  for(Start, End, Do) :-
18      call(Do, Start),
19      Next is Start + 1,
20      for(Next, End, Do).
21
22  % Write contador
23  writeContador(X) :- write(X), nl.
```

```
?- for(0,5, writeContador()).
0
1
2
3
4
true.
```

Instalando a IDE

Prolog possui diversas implementações, algumas delas são:

- Win-Prolog
- Ciao Prolog
- YAP Prolog
- SICStus Prolog
- SWI Prolog com SWI-Prolog-Editor
 - Mais recomendado
 - Obs.: SWI-Prolog-Editor somente disponível para windows

Instalando a IDE

SWI Prolog para windows ou MacOS

The image shows a Google search for "swi prolog". The search bar at the top contains "swi prolog". Below the search bar, there are tabs for "Todas", "Videos", "Imagens", "Noticias", "Shopping", "Mais", and "Ferramentas". The search results show approximately 486,000 results in 0.32 seconds.

The first search result is from [swi-prolog.org](https://www.swi-prolog.org). It includes a link to "Traduzir esta página" and a description: "SWI-Prolog offers a comprehensive free Prolog environment. Since its start in 1987, SWI-Prolog development has been driven by the needs of real world ...". Below this, there are links for "Download" (with sub-links for "Stable release", "Development release", and "Sources/building"), "SWISH" (with a link to "Use the Examples menu from the navigation bar"), and "Home download documentation" (with sub-links for "Did you know?", "Windows Release Notes", and "Building on Unix/POSIX"). There is also a link for "2.1 Getting started quickly" with a description: "A program swipl-win.exe, providing a window for ...".

The second search result is from [ufba.br](https://bruno.dac.ufba.br/aulas/swiprolog). It includes a link to "Traduzir esta página" and a title "Usando o SWI-Prolog como interpretador no Linux". The description says: "Esta página ensina como usar o SWI-Prolog como interpretador Prolog no Linux. Ela foi escrita para os alunos do BCC e não pretende ser uma fonte de consulta ...".

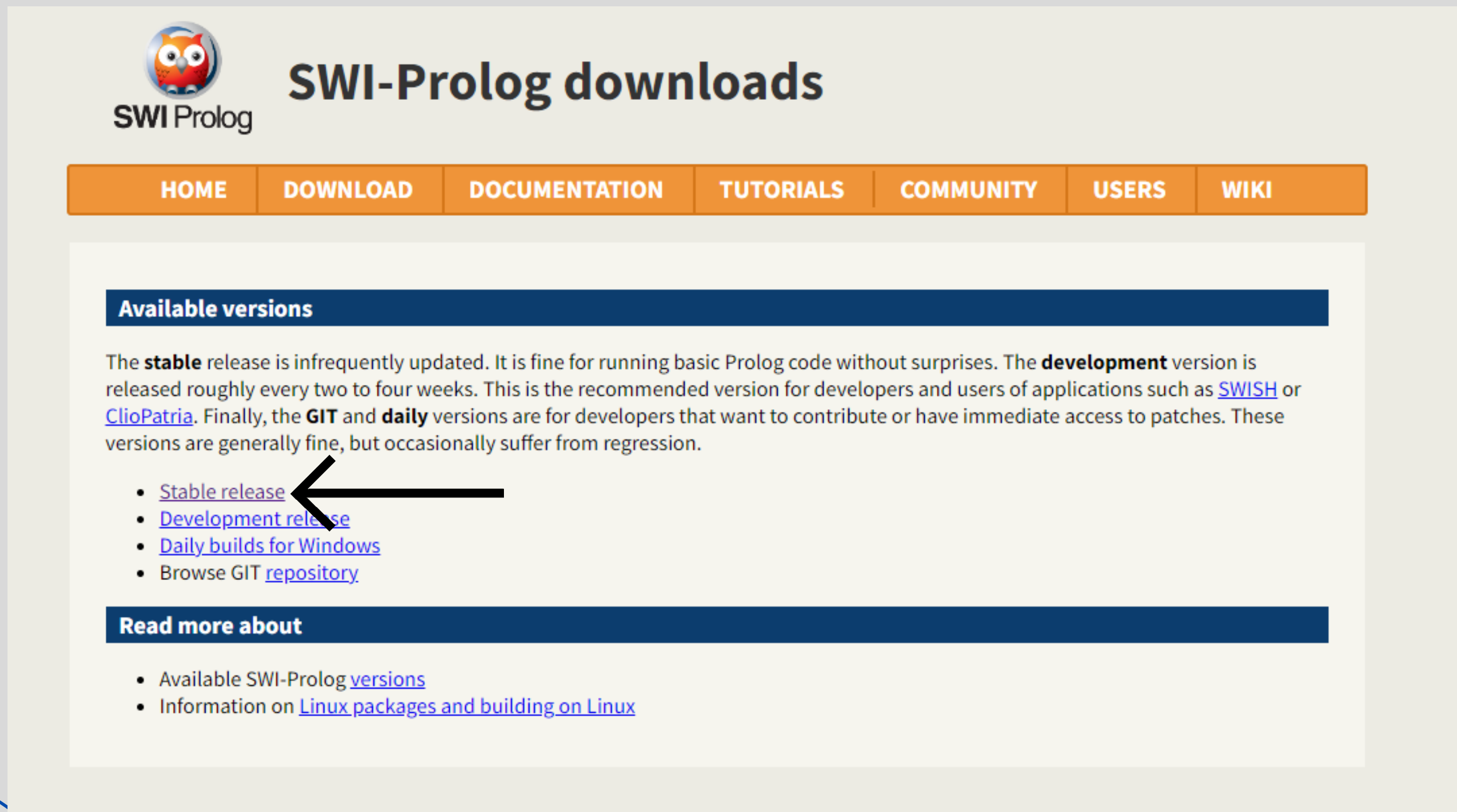
The third search result is from github.com. It includes a link to "Traduzir esta página" and a title "SWI-Prolog - GitHub". The description says: "Comprehensive Prolog compiler and development environment - SWI-Prolog."

The fourth search result is from [augustobaffa.pro.br](http://www.augustobaffa.pro.br/wiki/Como_executar_o_Prolog). It includes a link to "Traduzir esta página" and a title "Como executar o Prolog - Augusto Baffa Wiki".


On the right side of the search results, there is a knowledge panel for "SWI-Prolog". It features the SWI-Prolog logo (an owl) and a description: "SWI-Prolog é uma implementação em código aberto da linguagem de programação Prolog. Seu autor principal é Jan Wielemaker. Em desenvolvimento contínuo desde 1987, SWI-Prolog possui um rico conjunto de características, bibliotecas, ferramentas e uma documentação extensiva. Wikipédia". Below this, it lists "Autor original: Jan Wielemaker" and "Linguagens de programação: Prolog, C". At the bottom, there is a section "Itens também pesquisados" with links to "GNU Prolog", "Notepad++", "Scilab", and "WxMaxima".

Instalando a IDE

SWI Prolog para windows ou MacOS



The screenshot shows the 'SWI-Prolog downloads' page. At the top left is the SWI Prolog logo (an owl) and the text 'SWI Prolog'. To the right is the title 'SWI-Prolog downloads'. Below this is a navigation bar with orange buttons for 'HOME', 'DOWNLOAD', 'DOCUMENTATION', 'TUTORIALS', 'COMMUNITY', 'USERS', and 'WIKI'. The main content area has a dark blue header 'Available versions'. Below it, a paragraph explains the different release types: 'stable' (infrequently updated), 'development' (released every two to four weeks), and 'GIT' and 'daily' (for developers). A black arrow points to the 'Stable release' link in the list below. The list includes: 'Stable release', 'Development release', 'Daily builds for Windows', and 'Browse GIT repository'. Below this is another dark blue header 'Read more about' followed by two links: 'Available SWI-Prolog versions' and 'Information on Linux packages and building on Linux'.

 SWI Prolog

SWI-Prolog downloads

HOME DOWNLOAD DOCUMENTATION TUTORIALS COMMUNITY USERS WIKI

Available versions

The **stable** release is infrequently updated. It is fine for running basic Prolog code without surprises. The **development** version is released roughly every two to four weeks. This is the recommended version for developers and users of applications such as [SWISH](#) or [ClioPatria](#). Finally, the **GIT** and **daily** versions are for developers that want to contribute or have immediate access to patches. These versions are generally fine, but occasionally suffer from regression.

- [Stable release](#)
- [Development release](#)
- [Daily builds for Windows](#)
- Browse GIT [repository](#)

Read more about

- Available SWI-Prolog [versions](#)
- Information on [Linux packages and building on Linux](#)

Instalando a IDE

SWI Prolog para windows ou MacOS

HOME

DOWNLOAD


DOCUMENTATION


TUTORIALS


COMMUNITY


USERS







WIKI


Linux versions are often available as a package for your distribution. We collect information about available packages and issues for building on specific distros [here](#). We provide a [PPA](#) for [Ubuntu](#) and [snap images](#)


Android binaries are available for [Termux](#) as the package `swi-prolog`. See also [Building SWI-Prolog on Android using LinuxOnAndroid](#)



Please check the [windows release notes](#) (also in the SWI-Prolog startup menu of your installed version) for details.


Examine the [ChangeLog](#).

Binaries		
	13,163,366 bytes	SWI-Prolog 9.0.4-1 for Microsoft Windows (64 bit) Self-installing executable for Microsoft's Windows 64-bit editions. Requires at least windows 7. See the reference manual for deciding on whether to use the 32- or 64-bits version. This binary is linked against GMP 6.1.1 which is covered by the LGPL license. SHA256: 33758f1c2dd190df9c8828d2dcb39166ad10d31d78f1198812e6d0f33b71c73b
	13,203,365 bytes	SWI-Prolog 9.0.4-1 for Microsoft Windows (32 bit) Self-installing executable for MS-Windows. Requires at least Windows 7. Installs <code>swipl-win.exe</code> and <code>swipl.exe</code> . This binary is linked against GMP 6.1.1 which is covered by the LGPL license. SHA256: c99b7b794d14335ca6fda556f959e74c4b1b51877673a404f87c9cb68bce794c
	51,743,650 bytes	SWI-Prolog 9.0.4-1 for MacOSX 10.14 (Mojave) and later on x86_64 and arm64 Installer with binaries created using Macports . Installs <code>swipl</code> to <code>/usr/local/bin/swipl</code> . Needs xquartz (X11) and the Developer Tools (Xcode) installed for running the development tools . SHA256: a6f32683e4c42e62ea6f8f481ac1f5f5bfa2623b5c32eb213c1a04c5ebbc197
	28,195,489 bytes	SWI-Prolog 8.4.1-1 for MacOSX bundle on intel Installer with binaries created using Macports . Installs <code>swipl</code> to <code>/usr/local/bin/swipl</code> . Needs xquartz (X11) and the Developer Tools (Xcode) installed for running the development tools . SHA256: 1b9c62caa781818a0dafd1d822ab563b8c10c7cd018ce10a3b1f900eb3a434f
Sources		
	11,854,471 bytes	SWI-Prolog source for 9.0.4 Sources in <code>.tar.gz</code> format, including packages and generated documentation files. See build instructions . SHA256: feb2815a51d34fa81cb34e8149830405935a7e1d1c1950461239750baa8b49f0
Documentation		
	3,153,520 bytes	SWI-Prolog 9.0.4 reference manual in PDF SWI-Prolog reference manual as PDF file. This does <i>not</i> include the package documentation.
Show all files		


Instalando a IDE

SWI Prolog para windows ou MacOS

**SWI Prolog**

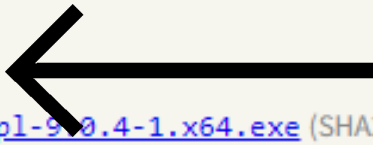
Download binary

[HOME](#) [DOWNLOAD](#) [DOCUMENTATION](#) [TUTORIALS](#) [COMMUNITY](#) [USERS](#) [WIKI](#)

 Windows antivirus software works using *signatures* and *heuristics*. Using the huge amount of virusses and malware known today, arbitrary executables are often [falsily classified as malicious](#). [Google Safe Browsing](#), used by most modern browsers, therefore often classifies our Windows binaries as malware. You can use e.g., [virustotal](#) to verify files with a large number of antivirus programs.

Our Windows binaries are cross-compiled on an isolated Linux container. The integrity of the binaries on the server is regularly verified by validating its SHA256 fingerprint.

Please select the checkbox below to enable the actual download link.

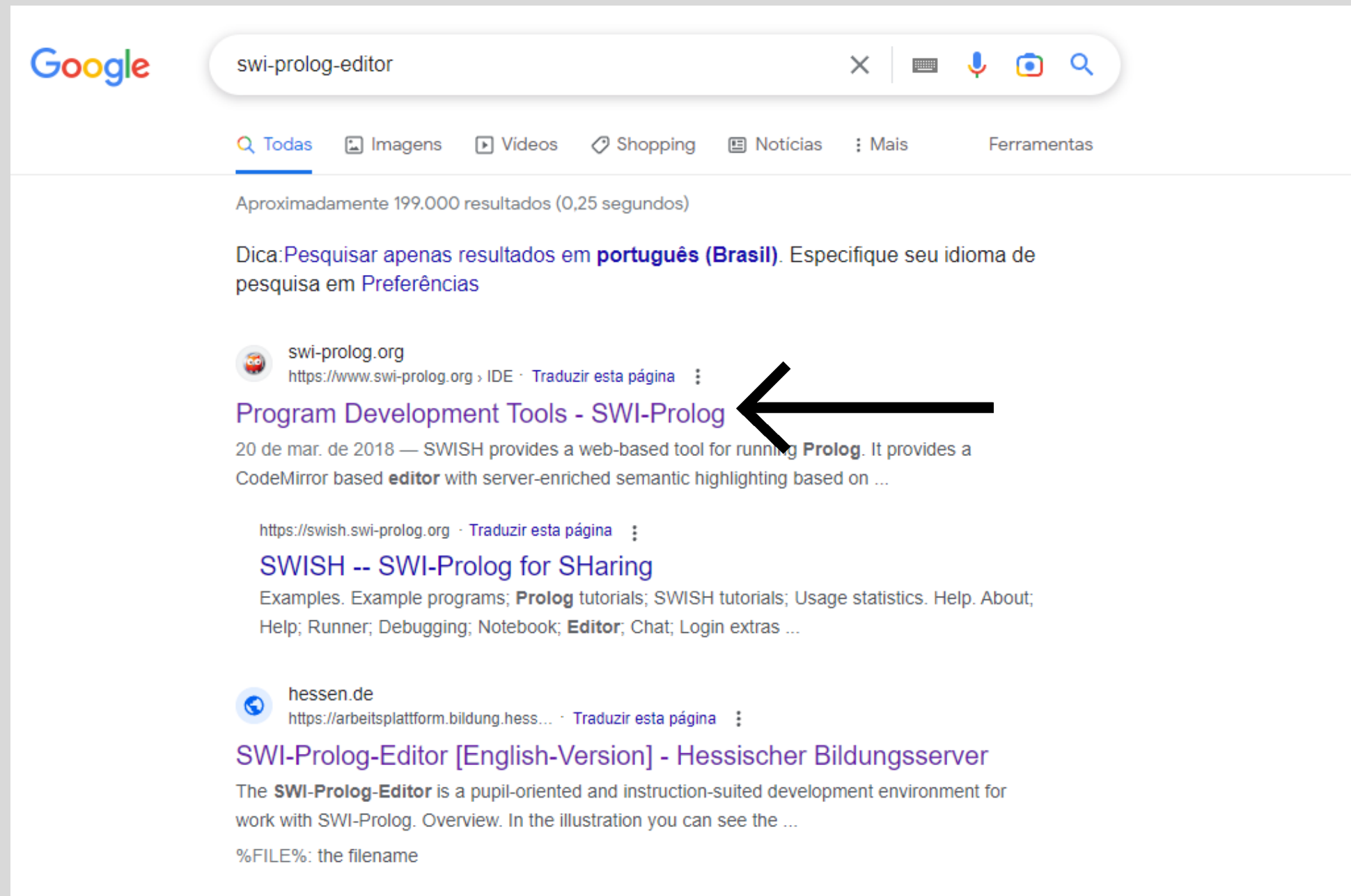
☐ I understand 

[Download swipl-9.0.4-1.x64.exe](#) (SHA256: 33758f1c2dd190df9c8828d2dcb39166ad10d31d78f1198812e6d0f33b71c73b)

[VIRUSTOTAL Scan Result](#)

Instalando a IDE

SWI-Prolog-Editor para windows



Instalando a IDE

SWI-Prolog-Editor para windows

- As a public server it can be used to run harmless queries against a Prolog itself or a read-only database.
- It can be installed with authentication enabled, which allows for running arbitrary Prolog programs.
- Mixed, where non-authenticated users can run harmless queries and authenticated users can run anything.

SWI-Prolog Editor (Windows)

Gerhard Röhner has developed an [integrated Prolog editor](#) for MS-Windows following the conventions of this platform. The embedded SWI-Prolog provides functionality similar to **swipl-win.exe**, including the possibility to run XPCE GUI programs.

Especially for classroom usage on MS-Windows, you should consider this version. The site also contains some demo material.

Support in standard editors

The lack of keywords, existence of dynamic operator declarations (see [op/3](#)), macro expansion and meta-calling make Prolog a difficult

Instalando a IDE

SWI-Prolog-Editor para windows

Installation of SWI-Prolog-Editor

The bit version of SWI-Prolog-Editor must be identically to the bit version of SWI-Prolog.

For personal usage you use this setup program:

- [32-bit version](#)
- [64-bit version](#)



For usage in a classroom-environment you do this: After the download of the SWI-Prolog-Editor in the

- [32-bit version](#)
- [64-bit version](#)



you unzip the files into a temporary folder und execute the setup.exe program.

The installation-program has special support for school-situations. It is possible to install the SWI-Prolog-Editor local or on a server. You can choose the folder for installation

Instalando a IDE

SWI Prolog para linux

Tarefa Simples:

- Abra o terminal do linux
 - Digite -> **sudo apt-get update**
 - Digite -> **sudo apt-get install swi-prolog**
 - Aguarde a instalação

Instalando a IDE

Alternativa para VScode

Extensão no VSCode:

- **Prolog**
- **VSC-Prolog**



 **Prolog** v0.0.4
Peng Lv | 138.307 | ★★★★★ (2)
Prolog language support for Visual Studio Code
Desabilitar ▾ Desinstalar ▾ ⚙️
Esta extensão foi habilitada globalmente.



 **VSC-Prolog** v0.8.23
arthurwang | 126.756 | ★★★★★ (14)
Support for Prolog language
Desabilitar ▾ Desinstalar ▾ ⚙️
Esta extensão foi habilitada globalmente.

Exemplos práticos

Árvore Genealógica

```
1  % Fatos representando pessoas
2
3  homem(joao).
4  homem(pedro).
5  homem(carlos).
6  homem(eduardo).
7
8  mulher(maria).
9  mulher(lucia).
10 mulher(ana).
11 mulher(julia).
12
13 %fatos relacionais entre pessoas
14 pai(joao, pedro).
15 pai(joao, lucia).
16 pai(pedro, ana).
17 pai(pedro, carlos).
18 mae(maria, pedro).
19 mae(maria, lucia).
20 mae(lucia, julia).
21 mae(lucia, eduardo).
22
23 % Regras para relacionar os gêneros
24 filho(X, Y) :- homem(X), pai(Y, X); homem(X), mae(Y, X).
25 filha(X, Y) :- mulher(X), pai(Y, X); mulher(X), mae(Y, X).
26 irmao(X, Y) :- homem(X), pai(Z, X), pai(Z, Y), X \= Y; homem(X), mae(Z, X), mae(Z, Y), X \= Y.
27 irma(X, Y) :- mulher(X), pai(Z, X), pai(Z, Y), X \= Y; mulher(X), mae(Z, X), mae(Z, Y), X \= Y.
```

Exemplos práticos

Quicksort

```
1 quicksort([], []).
2 quicksort([X|Xs], Ys) :-
3     partition(X, Xs, Ls, Rs),
4     quicksort(Ls, LsSorted),
5     quicksort(Rs, RsSorted),
6     append(LsSorted, [X|RsSorted], Ys).
7
8 partition(_, [], [], []).
9 partition(P, [X|Xs], [X|Ls], Rs) :- X @=< P, partition(P, Xs, Ls, Rs).
10 partition(P, [X|Xs], Ls, [X|Rs]) :- X @> P, partition(P, Xs, Ls, Rs).
```

Exemplos práticos

Movies.pl (parecido com o IMDB)



swipl/example/movies.pl

Trabalho Futuro

VII

Prolog Session

Chair: *Randy Hudson*
Discussant: *Jacques Cohen*

THE BIRTH OF PROLOG

Alain Colmerauer

Faculté des Sciences de Luminy
163 avenue de Luminy
13288 Marseille cedex 9, France

Philippe Roussel

Université de Nice-Sophia Antipolis, CNRS, Bat 4
250 Avenue Albert Einstein
06560 Valbonne, France

ABSTRACT

The programming language, Prolog, was born of a project aimed not at producing a programming language but at processing natural languages; in this case, French. The project gave rise to a preliminary version of Prolog at the end of 1971 and a more definitive version at the end of 1972. This article gives the history of this project and describes in detail the preliminary and then the final versions of Prolog. The authors also felt it appropriate to describe the Q-systems because it was a language that played a prominent part in Prolog's genesis.

CONTENTS

- 7.1 Introduction
- 7.2 Part I. The History
- 7.3 Part II. A Forerunner of Prolog, the Q-Systems
- 7.4 Part III. The Preliminary Prolog
- 7.5 Part IV. The Final Prolog
- Conclusion
- Bibliography

7.1 INTRODUCTION

As is well known, the name "Prolog" was invented in Marseilles in 1972. Philippe Roussel chose the name as an abbreviation for "PROgrammation en LOGique" to refer to the software tool designed to implement a man-machine communication system in natural language. It can be said that Prolog was the offspring of a successful marriage between natural language processing and automated theorem-

331

Considerações Finais

- Interessante conhecer o Paradigma Lógico
- Quebra da ideia de ser uma linguagem ultrapassada
- Diferentes possibilidades de aplicação
- Sintaxe relativamente simples

Bibliografia

- BEDREGAL, Benjamín; AICIÓLY, Benedito. Introdução à Lógica Clássica para a Ciência da Computação. **Universidade Federal do Rio Grande do Norte - UFRN**. Rio Grande do Norte, v.3.1, p. 205 - 238, jun., 2007. Disponível em: <https://www.dimap.ufrn.br/~jmarcos/books/BA_Jul07.pdf>. Acesso em: 31 mar. 2023.
- BOOTCAMP, Open. **¿Qué es el paradigma declarativo?**. Disponível em: <<https://open-bootcamp.com/aprender-programar/paradigma-declarativo>>. Acesso em 01 abr. 2023.
- COLMERAUER, Alan; ROUSSEL, Philippe. The birth of Prolog. **Prolog Session**, Marseille - França, p. 331-367, nov., 1992. Disponível em: <https://www.academia.edu/77294223/The_birth_of_Prolog>. Acesso em: 27 mar. 2023.
- COMO INSTALAR 2023. **Como instalar swi-prolog no Ubuntu**. Disponível em: <<https://howtoinstall.co/pt/swi-prolog>>. Acesso em: 22 mar. 2023.
- CLOCKSIN, William; Mellish, Christopher. **Programming in Prolog**. Londres - Inglaterra: Springer, 2003. Disponível em: <https://athena.ecs.csus.edu/~mei/logicp/Programming_in_Prolog.pdf>. Acesso em: 29 mar. 2023.

Bibliografia

- LAGO, Silvio. Introdução à Linguagem Prolog. Universidade de São Paulo - USP. Disponível em: <<https://www.ime.usp.br/~slago/slago-prolog.pdf>>. Acesso em: 25 mar. 2023.
- MEIDANIS, João. MC346 - Paradigmas de programação Prolog. **Universidade Estadual de Campinas - UNICAMP**. Disponível em: <<https://www.ic.unicamp.br/~meidanis/courses/mc346/2017s2/prolog/apostila-prolog.pdf>>. Acesso em 01 abri. 2023.
- NOLETO, Cairo. **Paradigmas de programação: o que são e quais os principais?**. Disponível em: <<https://blog.betrybe.com/tecnologia/paradigmas-de-programacao/>>. Acesso em 01 abr. 2023.
- SWI-PROLOG. **Robust, mature, free. Prolog for the real worldB**. Disponível em: <<https://www.swi-prolog.org/>>. Acesso em: 22 mar. 2023.
- VIEIRA, Leandro. **Faculdade de Tecnologia de São Paulo - FATEC**, São Paulo, 2015. Disponível em: <https://leandromoh.gitbooks.io/tcc-paradigmas-de-programacao/content/6_paradigma_logico/index.html>. Acesso em: 31 mar. 2023.