Ficha Prática 1

José Creissac Campos, António Nestor Ribeiro
{jose.campos, anr}@di.uminho.pt
PPIII (LESI)

2002/03

Objectivos

1. Instalar o SWI Prolog.
2. Aprender a trabalhar com o interpretador.
3. Fazer interrogações à informação existente.

Breve Introdução

Átomos, variáveis e números

Átomos podem ser construídos de três formas:

1. *strings* de letras, digítos e *underscores* começando por uma letra minúscula:

```
rui
mario
tcp4
t_3
t___
parad_prog_3
```

2. *strings* de caracteres especiais:

```
<...>
===>
..:..
:::::
```

3. *strings* de carateres entre pelicas:

```
'daytona'
'kafka'
```

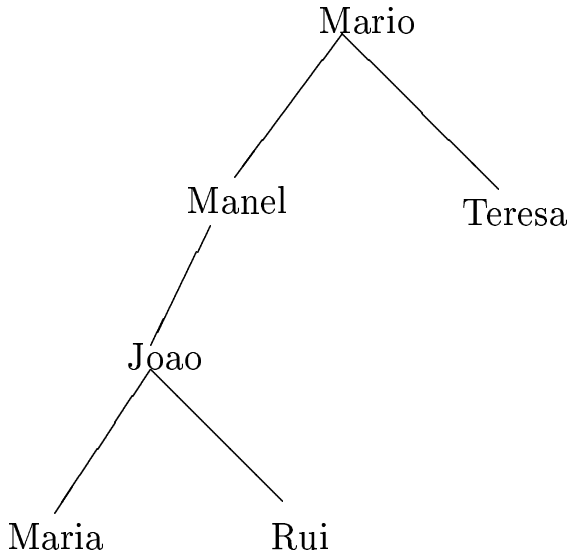
Variáveis são sequências de letras começadas por letra maiúscula ou por um underscore.

```
X
Y
_X
NomeVar
```

Números em Prolog incluem os números inteiros e os reais. No entanto os números reais não costumam ser muito utilizados, uma vez que o Prolog é antes de mais uma linguagem adequada à computação simbólica.

Exemplo - Base de Dados

Consideremos a seguinte árvore genealógica:



O conhecimento adquirido a partir da figura leva à existência dos seguintes ***factos***:

```
pai(joao, maria).
pai(joao, rui).
pai(manel, teresa).
pai(manel, joao).
pai(mario, manel).
```

Este conhecimento é aquele que é explícito a partir da figura. É agora possível fazer perguntas e obter mais conhecimento a partir da informação existente.

SWI Prolog

- Carregar um ficheiro

```
| ?- consult(bd).    %% bd corresponde ao nome do ficheiro sem extensão
```

ou,

```
| ?- [bd].
```

- Visualizar o conhecimento existente na base

```
| ?- listing.



pai(joao, maria).
pai(joao, rui).
pai(manel, joao).
pai(mario, manel).
pai(mario,teresa).

yes
```

- Sair do interpretador

```
| ?- halt.
```

Interrogações à base de dados

É possível interrogar o sistema de maneira a extrair conhecimento da base de dados. Por exemplo:

- Verificar se o `manel` é pai do `joao`.

```
| ?- pai(manel,joao).

yes
```

- Verificar se o `rui` é filho do `joao`.
- Verificar se a `teresa` é filha do `mario`.

Ou ainda fazer perguntas mais complexas como:

- Determinar os filhos do `manel`.

```
| ?- pai(manel,X).

X = teresa ? ;    %% ";" dá a resposta seguinte no caso de existir

X = joao ? ;

no
| ?-
```

- Quem é o pai da `teresa`?
- Os pares pai/filho existentes na base.
- Quem é o avô do `joao`.

```
| ?- pai(X,joao),pai(Y,X).

X = manel,
Y = mario ? ;   %% Y é o avô

no
```

  Repare que a ","significa a conjunção de termos. Tem a mesma semântica que o AND (ou $\wedge$).

- Verificar que o pai do `rui` é o mesmo que o pai da `maria`.

```
pai(X,rui),pai(X,maria).
```

Parte Prática

Instruções

1. Faça o *download* do SWIProlog (procure-o a partir de `http://sim.di.uminho.pt/Disciplinas/PPIII/LESI/`).

2. Instale o SWIProlog (se está a instalar em Windows, preste atenção à directoria de trabalho que define).

Exercícios

Socorro!

Experimentar os predicados `help` e `apropos`:

1. Depois de iniciar o interpretador, escreva a *query* "`apropos(help).`" (não esqueça o ponto — '.')
   Note que o predicado `apropos` lhe forneçe uma lista de predicados e secções do manual relacionados com o assunto indicado como parâmetro. (Se estiver a utilizar a versão 4.0 ou superior do SWI Prolog, essa lista aparacerá numa nova janela.)

2. Experimente agora a *query* "`help.`". (Se estiver a utilizar a versão 4.0 ou superior, poderá utilizar a área de diálogo que aparece na janela de *help*.)
   Note que o predicado `help` lhe fornece ajuda sobre os predicados indicados como parâmetro.

3. Experimente as *queries* "`help(1).`" e "`help(2-1).`".
   Note que quando o parâmetro do predicado `help` é um número, é apresentada a secção do manual com esse número. Na verdade tem todo o manual disponível *online* para consulta!

4. Finalmente, experimente a *query* "`help(halt).`".

Factos, *queries* e regras

1. Escreva os seguintes factos (pode utilizar "`[user].`" ou então escrever os factos num ficheiro e utilizar `consult/1`):

```
aluno(joao,ppi).
aluno(pedro,ppi).
aluno(maria,ppiii).
aluno(rui,ppiii).
aluno(manuel,ppiii).
aluno(pedro,ppiii).
aluno(rui,ppiv).
```

(a) Verifique que os factos estão presentes na Base de Conhecimento (utilize o predicado `listing`).

(b) Escreva uma *query* que verifique se `joao` é aluno de `ppiii`.

(c) Escreva uma *query* que verifique se `rui` é aluno de `ppi` — note o efeito do Princípio do Mundo Fechado.

(d) Escreva uma *query* que verifique se `joao` e `maria` são ambos alunos de `ppiv` — `joao` e `maria` são ambos alunos de `ppiv` se `joao` for aluno de `ppiv` e `maria` for aluna de `ppiv`.

(e) Escreva uma *query* que permita saber quem é aluno de `ppiii`.

(f) Escreva uma *query* que permita saber de que disciplinas é `rui` aluno.

2. Adicione os seguintes factos à base de conhecimento (se anteriormente utilizou "`[user].`" é agora uma boa altura para começar a utilizar ficheiros):

```
estuda(joao).
estuda(maria).
estuda(manuel).
```

(a) Sabendo que a aluno A faz a disciplina P se A é aluno de P e A estuda, escreva uma *query* que lhe permita saber se `maria` vai fazer `ppiii`.

(b) Experimente agora a *query* "`aluno(X,ppiii),estuda(X).`". O que lhe permite esta *query* saber?

(c) Utilizando a *query* da alínea anterior, acrescente à Base de Conhecimento o predicado `fazppiii/1` e escreva uma *query* que lhe permita saber quem faz `ppiii` (não se esqueça de fazer `consult` do ficheiro).