Inteligência Artificial

2018/2019

Exercícios de programação em lógica

1. Dada uma base de dados com os seguintes predicados (ver p1.pl):

   - homem(*nome*, *profissão*, *data de nascimento*)
   - mulher(*nome*, *profissão*, *data de nascimento*)
   - filho(*nome*,*nomeDoPaiOuDaMae*)

2. Defina o predicado: irmao(*nome1*,*nome2*)

   Tenha em conta que uma pessoa não é irmã de si mesma.

3. Defina o predicado: primoDireito(*nome1*,*nome2*)

   Primos direitos são os filhos dos irmãos do pai ou da mãe.

4. Defina o predicado: irmãos(*nome1*,-[*nome**])

   Dado um nome *nome1* deve construir uma lista com todos os irmãos de *nome1*. (sugestão: use o predicado *findall*)

5. Defina o predicado: primos(*nome1*,-[*nome**])

   Primos são: os primos direitos, os filhos dos primos e os primos dos pais.

6. Defina o predicado: esposa(*nomeMulher*,*nomeHomem*)

   Considere que esposa de um homem é uma mulher que é mãe do filho do homem.

7. Defina o predicado: descende(*nome*, -[*ascendentes*])

   Dado um nome deve construir uma lista com todos os ascendentes na base de dados, pais, avós, bisavós, etc.

8. Defina o predicado: descendentes(*nome*, -[*descendentes*])

   Dado um nome deve construir uma lista com todos os descendentes, filhos, netos, bisnetos, etc.

9. Defina o predicado: ascendentes(*nome*, c(Nome,AscendentesPai,AscendentesMae)).

   ex:

```
ascendentes('Sancho II',L).

L = c('Sancho II',
          c('Afonso II',
              c('Sancho I',
                  c('Afonso Henriques',
                      c('Henrique de Borgonha',0,0),
```

```
                    c('Teresa de Castela',0,0)),
                c('Mafalda',0,0)),
            c('Dulce de Barcelona',0,0)),
        c('Urraca C',0,0))
```

10. Defina o predicado: descendentes(*Nome*, c(*Nome*, *NomeEsposa*, -[*DescendentesFilho**])).
    ex:

```
descendentes('Afonso Henriques',L).

L = c('Afonso Henriques','Mafalda',
          [c('Fernando II','Urraca',[c('Afonso IX',0,[])]),
           c('Sancho I','Dulce de Barcelona',
                    [c('Afonso II','Urraca C',
                          [c('Sancho II',0,[]),c('Afonso III',0,[])]),
                           c(0,'Bereng?ria',[])])])
```

11. Defina o predicado: descendentesNivel(*Nome*, Nivel, [Nomes])
    Se Nível igual a 1 lista os filhos
    Se Nível igual a 2 lista os netos
    Se Nível igual a 3 lista os bisnetos
    etc.
    Deve utilizar o predicado já definido descendentes.
12. Defina o predicado: ascendentesNivel(*Nome*, Nível, [Nomes])
13. Defina o predicado: ordenaNomesAlf(+lista,-listaordenada)
    Que dada uma lista de nomes constrói uma lista com os nomes ordenados por ordem alfabética crescente.
    Use o insertion sort.
14. Defina o predicado: ordenaNomesIdade(+lista,-listaordenada)
    Que dada uma lista de nomes constrói uma lista com os nomes ordenados por ordem de nascimento crescente.
    Use o merge sort.
15. Complete a base de dados incluindo informação até ao fim da primeira dinastia *D João I, mestre de Aviz*.
16. Defina o predicado *troca(+lista,p1,p2,-listaTrocada* que dada uma lista, constroi a listaTrocada que tem os mesmo elementos da lista excepto os elemento que estão nas posições p1 e p2 que devem trocar de posição.