Tópicos em Inteligência Artificial

Prolog

UFFS
Ciência da Computação

09/05/2017

Prova 1

Nome: _____ Matrícula: _____

Prova 1: [130 pontos / 100]

Instruções:

1. A prova será feita nos computadores do Laboratório. Nada de laptop ou computadores pessoais.
2. Coloque seu nome e matrícula no arquivo de repostas.
3. Leia bem a prova. A leitura e interpretação das questões é responsabilidade do aluno.
4. Caso o computador que você está usando não tenha algum software instalado, instale ou solicite para trocar de computador.
5. A prova é com consulta ao material individual do aluno. Não será permitido empréstimo de material. Materiais permitidos: Livros, apostila, resumo, programas.
6. *Não é permitido o uso de internet.* Alunos pegos usando a internet terão nota Zero.
7. Ao final da prova chame o professor para colocar o arquivo resposta no pen drive.
8. Para obter nota 10,0 (Dez) o aluno precisa de 100 pontos. Os pontos extras são para ajudar os alunos. **No entanto a nota máxima continua sendo 10,0**.

Duração da prova: 2 horas e 10 minutos

Boa prova!

Questões:

1. [50 **pts**] Construa uma base de dados que modele um campeonato de tênis e faça predicados para recuperar informações nesta base de dados. Pelo menos as seguintes informações devem estar presentes:

- Campeonatos: com tipo (Simples masculino, Simples Feminino, Duplas mistas, ...), e data.
- Participantes: com campeonato, ordem de inscrição, nome, gênero, país que representa e data de nascimento.
- Jogos: com campeonato, fase do jogo, adversários, resultado e vencedores.
- Campeão: com campeonato, nome do(a) campeã(o).

(a) [10 **pts**] Representação: Obs: não precisa colocar todos os dados mas apenas alguns exemplos, no entanto todos os tipos de predicados devem ter pelo menos alguns exemplo para que a execução das questões abaixo possa ser realizada sobre a base de dados.

(b) Faça predicados para recuperar as seguintes informações:

1. [10 **pts**] Qual a tenista mais velha da competição?
2. [10 **pts**] Quem venceu mais campeonatos dentre os cadastrados?
3. [10 **pts**] Uma lista com os adversários de um participante num campeonato especifico.
4. [10 **pts**] Quantos jogos um participante jogou num tipo de campeonato, independente do ano do evento?

Solução:

```
campeonato(11, s/m,inicio(12/02/2005),fim(30/02/2005)).
participante(10, 16,'Martina Navratilova', EUA, f, nasc(1956/09/18)).
participante(10, 01,'Serena Williams', EUA, f, nasc(1981/10/20)).
jogo(10,c,data(15/02/2005),16, 01,res(2/1), 16). % fases: classificação,1 fase,
jogo(10,c,data(17/02/2005),05, 08,res(0/2), 08). % semi-finais, final.
campeao(10, 'Martina Navratílova').
campeao(05, 'Serena Williams').
campeao(02, 'Serena Williams').


mais_velha(X):- findall(N, participante(_,_,N,_,f,_), L), acha_mais_velha(L,X),
acha_mais_velha([X], X) :- !.
acha_mais_velha([X|R], Y):- acha_mais_velha(R,Z),
        pnasc(Z,D1), pnasc(X,D2),
        (D1 >= D2, Y = X | Y = X), !.

pnasc(X,D) :- participante(_,_,X,_,_,nasc(D)), !.

mais_campeonatos(X):- findall(N, campeao(_,N), L),
        list_to_set(L,L1), maior_campeao(L1,X), !.
```

```
maior_campeao([X], X):- !.
maior_campeao([X|R], Y):- maior_campeao(R,Z),
         findall(C, campeao(C,Z), L1),
         findall(C1, campeao(C1,X), L2),
         length(L1,T1), length(L2,T2),
         (T1 >= T2, Y = Z | Y = X), !.

adversarios(X,Y,L1) :- participante(Y, I, X, _,_,_),
          findall(N, jogo(Y, _, _, I, N, _,_), L),
     pega_nome(Y,L,L1), !.

pega_nome(Y,[], []) :- !.
pega_nome(Y,[X|R1], [N|R2]) :- participante(Y,X,N,_,_,_),
          pega_nome(Y,R1,R2), !.


jogos_camp(X,Y,N) :- participante(Y, I, X, _,_,_),
          findall(D1, jogo(Y, _, D1, I, _, _,_), L1),
          findall(D2, jogo(Y, _, D2, _, I, _,_), L2),
     length(L1,T1), length(L2,T2), N is T1 + T2, !.
```

2. [50 **pts**] Represente em Prolog os seguintes predicados sobre listas sem usar os predicados pré-definidos de listas do Prolog.

(a) [10 **pts**] Ordena uma lista dada e retorna a lista ordenada.

(b) [10 **pts**] Zera todos os valores negativos de uma lista dada e retorna a lista atualizada.

(c) [10 **pts**] Recebe duas listas ordenadas e cria uma lista tambem ordenada com todos os itens das listas dadas

(d) [10 **pts**] Recebe uma lista e conta quantos itens estão duplicados na lista e retorna este valor.

(e) [10 **pts**] Recebe uma lista e retorna o maior valor da lista.

Solução:

```
ordena(L,L1) :- ordena(L,[],L1), !.

ordena([], L,L) :- !.
ordena([X|R], L1,L2) :- inclui_ordem(L1,X,L3),
           ordena(R,L3,L2), !.

inclui_ordem([], X, [X]) :- !.
inclui_ordem([Y|R], X, [X,Y|R]) :- X @< Y, !.
inclui_ordem([Y|R1], X, [Y|R2]) :- inclui_ordem(R1,X,R2),  !.

zera([], []) :- !.
```

```
zera([X|R1], [0|R2]) :- X < 0, zera(R1,R2), !.
zera([X|R1], [X|R2]) :- zera(R1,R2), !.

mistura([], [], []) :- !.
mistura([X|R], [], [X|R]) :- !.
mistura([], [Y|R], [Y|R]) :- !.
mistura([X|R1], [Y|R2], [X|R3]) :- X < Y,
    mistura(R1,[Y|R2],R3), !.
mistura([X|R1], [Y|R2], [Y|R3]) :- mistura([X|R1],R2,R3), !.

duplicados([], 0) :-  !.
duplicados([X|R], N) :- member(X,R), removetodos(X,R,R1),
         duplicados(R1,N1), N is N1 + 1, !.
duplicados([X|R], N) :- duplicados(R,N), !.

removetodos(X,[],[]) :- !.
removetodos(X,[X|R1],R2) :- removetodos(X,R1,R2), !.
removetodos(X,[Y|R1],[Y|R2]) :- removetodos(X,R1,R2), !.

maior([X],X) :- !.
maior([X|R1], Z) :- maior(R1,Y), (X >= Y, Z = X | Z = Y), !.
```

3. [30 **pts**] Dado o banco de dados abaixo.

```
% filial(<Código>,<Nome>)
filial(1, 'POA').
filial(2, 'Curitiba').

% cliente(<Código>,<filial>,<Nome>,<idade>,<Profissão>)
cliente(2345, 1, 'Carlos', 45, 'Médico').
cliente(2346, 2, 'Rogério', 23, 'Estudante').
cliente(2347, 2, 'Marcelo', 52, 'Advogado').
cliente(2348, 1, 'Ana', 35, 'Modelo').
cliente(3568, 1, 'Marcos', 20, 'Operário').
cliente(3569, 2, 'José', 45, 'Professor').
cliente(3570, 2, 'João Carlos', 38, 'Médico').

% item(<Código>,<Tipo m/c>,<Modelo>,<preço>).
item(101, m, 'Kawasaki 1000', 24000).
item(102, c, 'Audi A4', 57000).
item(103, c, 'BMW Serie 3', 42000).
item(107, c, 'Ford Focus', 25000).
item(112, c, 'Fiat Uno', 20000).
item(115, c, 'Fiat Pálio 1.6', 30000).
item(099, m, 'CB 450', 15000).
item(113, c, 'Ford Fox', 35000).
item(117, c, 'Mercedes', 80000).
```

```
% compra(<Cliente>,<item>,<data com dia,mês e ano>).
compra(2345,117,data(21,02,2017)).
compra(2346,101,data(23,03,2015)).
compra(3570,099,data(02,01,2015)).
compra(2348,103,data(10,04,2016)).
compra(2347,102,data(10,01,2017)).
compra(3568,112,data(04,07,2015)).
compra(2347,107,data(05,03,2016)).
compra(3569,115,data(05,03,2016)).
compra(3570,113,data(05,07,2016)).
```

Use os predicados de coleta de soluções para criar os predicados abaixo:

(a) [10 **pts**] Lista todos os bens de um cliente especificado pelo nome.

(b) [10 **pts**] Lista todos os bens dos clientes que tem uma determinada profissão.

(c) [10 **pts**] Lista todos os bens comprados em ordem de data de compra.

Solução:

```
lista_bens(N,L2) :- cliente(C,_,N,_,_),
       findall(B, compra(C,B,_),L1),
       pega_nome_item(L1,L2), !.

pega_nome_item([],[]) :- !.
pega_nome_item([C|R1],[N|R2]) :- item(C,_,N,_),
      pega_nome_item(R1,R2), !.


lista_bens2(N,L3) :- findall(C, cliente(_,_,C,_,N), L),
      pega_bens(L,L1), flatten(L1,L2), list_to_set(L2,L3), !.

pega_bens([], []) :- !.
pega_bens([X|R1], [L|R2]) :- lista_bens(X,L),
     pega_bens(R1,R2), !.

bens_ordem(L4):- findall((D,B), compra(_,B,D), L),
      arruma_data(L,L1), ordena(L1, L2),      separa_item(L2,L3), pega_nome_ite

arruma_data([],[]) :- !.
arruma_data([(data(D,M,A),B) | R1],
            [(A/M/D, B) | R2]) :- arruma_data(R1,R2), !.

separa_item([],[]):-!.
separa_item([(_,B)|R1], [B|R2]):-  separa_item(R1,R2), !.
```

Boa Prova