PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO – 2019-1

Profa. Heloisa - Primeiro Trabalho - **DATA DE ENTREGA: 16/05/2019**

==========

Escreva um programa Prolog para, dadas duas listas *L1 e L2* que contém elementos de qualquer tipo, possivelmente com repetições e com sublistas, construir outra lista que mostre quantas vezes cada elemento atômico (átomo, número, lista vazia) ou estrutura, que não seja lista, aparece **apenas em uma das duas listas dadas**, inclusive nas sublistas. A lista resultante deve conter pares de elementos (lista de dois elementos) sendo o primeiro elemento do par um elemento que só aparece em uma das duas listas dadas e o segundo elemento do par, o número de vezes que esse elemento aparece na lista. Os elementos que aparecem nas duas listas não devem ser contados e não aparecem na lista final (ver exemplo dos átomos b, número 5, e estrutura par(c,d) nas listas abaixo). Variáveis também devem ser descartadas (ver exemplo das variáveis Z e F nas listas abaixo). Os átomos devem aparecer na lista resultante na mesma sequência em que aparecem nas listas originais: primeiro os da primeira lista e depois os da segunda lista.

Por exemplo, dadas as listas:

L1 = [a, b, Z, [a, x], [5,x], [x], par(c,d)] e L2 = [4.6, 5, w, [], F, b, [], par(c,d), [], par(1,2)]

Deve ser construída a lista

Lout = [[a, 2], [x, 3], [4.6, 1], [w,1], [[], 3], [par(1,2), 1]].

Sugestão para solução:

Desparentizar as duas listas (construir listas sem sublistas), remover as variáveis e remover os elementos que aparecem nas duas listas. Depois, fazer o *append* dessas duas listas e construir a lista de pares.

Predicados obrigatórios e opcionais:

- Um predicado *conta_atomos(L1, L2, Lout)* que executa a operação solicitada (obrigatório);
- Um predicado principal para ler as duas listas dadas do teclado, e chamar o predicado *conta_atomos(L1,L2,Lout)*, que faz a operação principal. A lista resultante pode ser mostrada de qualquer forma (como variável na consulta ou com write) (obrigatório);
- Um predicado *desparentize(L, Lout)* que, dada uma lista *L*, constrói uma lista *Lout* sem sublistas, com apenas os elementos atômicos (incluindo lista vazia) e estruturas que aparecem em todos os níveis, descartando as variáveis. As variáveis podem ser removidas pelo próprio predicado *desparentize* ou podem ser removidas depois, por outro predicado (opcional);
- Um predicado *tira_comuns(L1,L2,L)* que, dadas duas listas *L1* e *L2*, retira de *L1* e de L2 os elementos que aparecem nas duas listas. (opcional);
- Um predicado *monta_pares(Lin, Lout)* que, dada uma lista de elementos *Lin*, sem sublistas, monta uma lista de pares formados por um elemento da lista *Lin* e o número de vezes que ele aparece em *Lin*;
- Outros predicados que achar necessário, se desejar subdividir as etapas da operação ou melhorar a entrada e saída.

Lembretes:

- Uma estrutura é um objeto composto que não é lista;
- Lista vazia é considerada tanto átomo como lista.

Observações:

- Podem ser utilizados, se necessário, os predicados pré-definidos de Prolog is_list(L), atomic(X), var(X), compound(X), not(G), append(L1, L2,Lout), member(X,L), write(X), read(X) ;
- Os trabalhos podem ser feitos em duplas, que deverão ser as mesmas em todos os trabalhos;
- Usar OBRIGATORIAMENTE, SWI-PROLOG para implementar o trabalho;
- Entregar, no AVA da disciplina (tarefa com envio de arquivo único):
  - Relatório (Arquivo txt, doc ou pdf) com listagem do código fonte, explicação do funcionamento de cada um dos predicados definidos e resultados de execução de pelo menos dois exemplos, sendo um deles o exemplo deste enunciado;
  - Arquivo do prolog com o código fonte do trabalho.

- **DATA DE ENTREGA: 16/05/2019**