Lista de Exercícios

Lógica em Programação

1) Considere um grafo com custos associados aos arcos que conectam os nós. Escreva um programa Prolog para encontrar o custo mínimo para ir de um nó A para um nó B do grafo.

2) Defina o predicado `height(BinaryTree,Height)` para calcular a altura de uma árvore binária. Assuma que a altura da árvore vazia é 0, e que a altura de uma árvore de um elemento é 1.

3) Defina um predicado para reconhecer se um termo Prolog é uma lista.

4) Os predicados `sub1`, `sub2` e `sub3`, abaixo, implementam uma relação de listas. O predicado `sub1` tem uma definição mais procedural do que os predicados `sub2` e `sub3`. Estes dois últimos são escritos em estilo mais declarativo. Estude o comportamento, no que diz respeito à eficiência, destes três predicados. Dois deles têm mais ou menos a mesma eficiência. Quais deles? Por que o outro é mais ineficiente?

```
sub1(List,Sublist) :- prefix(List,Sublist).
sub1([_|Tail],Sublist) :- sub1(Tail,Sublist).

prefix(_,[]).
prefix([X|List1],[X|List2]) :- prefix(List1,List2).

sub2(List,Sublist) :- conc(List1,List2,List),
                      conc(List3,Sublist,List1).

sub3(List,Sublist) :- conc(List1,List2,List),
                      conc(Sublist,_,List2).
```

5) Defina a relação `reverse(List,RevList)`, onde os argumentos são representados por diferença de listas.

6) Utilize o predicado **bagof/3** para definir a relação `powerset(Set,Subsets)` que compute o conjunto de todos os subconjuntos de um determinado conjunto. (represente todos os conjuntos por listas).

7) Defina a relação `alv(Tree)` para testar se uma árvore binária é AVL, isto é, todas subárvores devem diferir em altura de no máximo 1. Represente a árvore binária através do termo: `t(Left,Root,Right)` ou `nil` se a subárvore estiver vazia.

8) Como é que os programas de busca dados em aula poderiam ser modificados para começar a busca a partir de um conjunto de estados iniciais, ao invés de somente um estado inicial?