LISTA 1 - Prolog
CURSO: GameAI

1. Escreva um programa em PROLOG que represente o conhecimento sobre literatura do seguinte tipo:

```
tipo(titulo,tipoDaObra)
escritoPor(titulo,pessoa)
escritor(pessoa)
publicado(titulo,ano)
```

onde `titulo` e `pessoa` são listas, por exemplo `[from,whom,the,bells,tolls]` e `[machado,de,assis]`, e `tipoDaObra` pode ser romance ou teatro.

Defina também alguns conceitos, por exemplo:

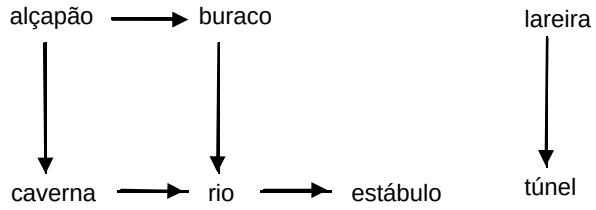
[1] "um título é classificado como ficção se ele é um romance ou uma peça-de-teatro"
[2] "uma pessoa é romancista se ela é um escritor cujos títulos são todos romances"
[3] "uma pessoa é um autor moderno se ela é um escritor cujos títulos são todos publicados após 1900"

Teste os conceitos acima e faça as seguintes perguntas compostas:

Existe algum romance escrito por … ?
Há romancistas modernos ?

OBS: A única dificuldade deste exercício está no TODOS dos conceitos [2] e [3]. No caso de [2], a solução mais simples é interpretar [2] como sendo apenas um predicado `eh_romancista(X)` que só é utilizado quando `X` está instanciado (i.e. não serve para extrair todos os romancistas, serve apenas para testar se alguém é ou não romancista). Neste caso, o problema se resolve com negação por falha (not). Casos mais gerais precisam de vários predicados auxiliares (e.g. autorVariosEstilos e autorVariasObras) e do uso do not. Mesmo assim, estas soluções tendem a gerar repetições à pergunta `romancista(X)`. Uma boa solução geral pode ser construída com o predicado extra-lógico `forall`. Outra solução geral pode utilizar o predicado extra-lógico `findall` que coleta todas as respostas possíveis (repetidas ou não) em uma lista (neste caso seria interessante eliminar as repetições). As mesmas observações valem para [3].

2. Considere que uma estratégia de jogo é representada por um grafo direcionado de locais através de uma coleção de fatos `edge(Node1,Node2)`. Escreva um programa em Prolog que defina a relação `connected(Node1,Node2)`, a qual é verdade se `Node1` e `Node2` estão conectados. Note que se trata de estabelecer o fecho transitivo da relação `edge`. Note que conceitualmente um nó está conectado a ele mesmo. Use o exemplo abaixo para provar que `connected(alcapao,estabulo)` é verdade e verificar que `connected(rio,lareira)` não consegue ser provado.



OBS: A solução para um grafo não- direcionado é muito mais complexa.. A solução para o caso direcionado são apenas duas cláusulas muito simples.

3. Escreva em Prolog as relações `sufixo(S,L)`, onde `S` é uma sublista terminal de `L`, e `prefixo(P,L)`, onde `P` é uma sublista inicial de `L`. Por exemplo, `sufixo([bells,tolls],[from,whom,the,bells,tolls])`, e `prefixo([from,whom],[from,whom,the,bells,tolls])` são verdades.

OBS: não defina sufixo em função de prefixo e vice-versa.