SISTEMAS INTELIGENTES

PROF. TACLA

OBJETIVO:

Compreender PROLOG como uma linguagem declarativa e procedural com o jogo Fox and Ducks.

PARA FAZER

- ler o artigo (Merrit, 1993) Merrit, D.C. *Exploring Prolog: Adventure, Objects, Animals, and Taxes,* PC AI Magazine, October 1993.
- Implemente o código que está descrito na seção Adventure game do artigo acima.
- Faça os exercícios abaixo e carregue o código resultante no moodle

EXERCÍCIO 1: observe que ao executar o jogo pela 2a. vez ele já entrará no estado final. Os objetos se encontram num estado que satisfaz o predicado done. Faça um 'procedimento' de reinicialização de forma que ao final do jogo os objetos retornem ao estado inicial.

EXERCÍCIO 2: transforme o predicado connect em uma relação reflexiva e simétrica de forma que só seja necessário indicar as conexões em um sentido por meio do predicado connect(X,Y).

EXERCÍCIO 3: faça com que a raposa ande aleatoriamente no jogo. Inclua um contador para saber quantos patos a raposa comeu.

DICAS DE SINTAXE E SEMÂNTICA

Dyamic: Inclua no início do seu código a declaração *dynamic* para indicar quais predicados serão modificados com delete (apagar) e assert (inserir) em tempo de execução. O '/1' significa que o predicado possui um argumento '/2', dois, e assim sucessivamente.

```
:- dynamic you_have/1, location/2.
```

Write: para escrever algo na tela utilize write com aspas duplas ou simples (pode variar de acordo com a versão do Prolog). Por exemplo:

```
write("You are in the "), writeln(X).
ou
write('You are in the '), writeln(X).
```

read(X): faz leitura do teclado e armazena em X. Por exemplo, se o usuário teclar *goto(yard),* a variável X será igual *goto(yard)*.

call(X): invoca o predicado que está armazenado na variável X. Por exemplo, se X = *goto(duck_pen)*, Prolog tentará unificar a cláusula goto(duck_pen) com a cabeça de alguma cláusula da base de conhecimentos.