Paradigmas da Programação I
LECom (2º ano)

Trabalho Prático nº 2

Ano lectivo 2004/2005

1 Objectivos e Organização

Este trabalho prático tem como principais **objectivos**:

- aumentar a experiência de uso de uma linguagem (declarativa) lógica para modelar estratégias de decisão e de planeamento de acções;
- praticar o uso de *listas* e da *recursividade*;
- aumentar a experiência de programação na linguagem lógica Prolog para resolução geral de problemas, encontrando a resposta a questões formuladas (num determinado universo de discurso) ao construir a árvore de prova que verifica se a interrogação colocada faz parte, ou se se pode inferir, da teoria (programa) apresentada;
- aperfeiçoar o uso do ambiente de programação lógica SWI-Prolog.

Além disso é intenção desta folha de trabalho, estimular nos alunos o gosto pelos jogos tradicionais e sua programação e prepará-los para concursos de programação.

Para o efeito, esta folha contém 2 grupos com um total de 7 enunciados, devendo resolver dois, um de cada grupo.

O trabalho–a realizar até ao final do mês de Dezembro–ou seja, o programa desenvolvido, deve ser submetido automaticamente por cada grupo mal esteja concluído, usando para isso o sistema de submissão electrónica disponibilizado na página da disciplina. Para efeito de avaliação, será depois (em data a combinar, mas fora da aula) apresentado ao docente a funcionar (acompanhado do respectivo relatório de desenvolvimento) e defendido pelo grupo (3 alunos).

O **relatório** a elaborar deve ser claro e, além do respectivo enunciado, da descrição do problema, e das escolhas/decisões que determinaram a sua implementação final, deverá conter exemplos de utilização e o código do programa. Como é de tradição, o relatório será escrito em LaTeX.

2 Grupo 1 de Enunciados

2.1 Torres de Hanoi

Desenvolva um programa em Prolog que implemente o famoso jogo das Torres de Hanoi em que se pretende mover os `n` disco empilhados na Torre A (por ordem decrescente do seu diâmetro) para a Torre C, usando a Torre B como auxiliar. De cada vez só pode mover 1 disco de uma torre para a outra, mas jamais pode colocar um disco sobre outro de menor diâmetro.

O programa deve ser invocado através do predicado `torresHanoi/1`, em que o argumento único é o número de disco com que se pretende jogar. O programa escreverá cada movimento a efectuar indicado a torre origem e a torre destino.

2.2 8 Rainhas

Desenvolva um programa em Prolog que implemente o famoso quebra-cabeças das 8 Rainhas cujo objectivo é colocar 8 rainhas num tabuleiro de xadrez `8x8` de tal modo que nenhuma fique em xeque[1].
O programa deve ser invocado através do predicado `oitoRainhas/0`. Ao terminar, o programa indica em que posição (Linha, Coluna) se deveria colocar cada Rainha, ou então sinalizar que não existe solução.
Sugestão: altere o predicado inicial de modo a receber como parâmetro a dimensão do tabuleiro (e consequentemente o número de Rainhas a colocar).

2.3 Saltos de Cavalo

Desenvolva um programa em Prolog que implemente o famoso quebra-cabeças das Saltos de Cavalo cujo objectivo é atravessar um tabuleiro de xadrez `NxN` a partir de uma casa dada, com saltos de cavalo[2] de tal modo que todas sejam visitadas 1 e 1 só vez.
O programa deve ser invocado através do predicado `saltosCavalo/3`, em que o primeiro argumento é a dimensão do tabuleiro e os dois seguintes a coordenada da casa de partida.
Ao terminar, o programa indicará o percurso feito pelo Cavalo no tabuleiro, registando o número do salto em que pousa em cada casa.

3 Grupo 2 de Enunciados

3.1 Jogo da Forca

Neste trabalho, pretende-se que crie desenvolva em Prolog um programa para jogar com o Utilizador uma versão mono-palavra do conhecido Jogo da Forca, em que o computador escolhe uma palavra-mistério e o jogador tem `n` tentativas para a adivinhar, propondo uma letra de cada vez.
O programa deve ser invocado através do predicado `jogoForca/1`, em que o argumento único é o número de tentativas permitidas. Ao começar, o computador mostra a palavra-mistério com uma marca qualquer na posição de cada caracter. Em cada jogada o computador lê uma letra e volta a mostrar a palavra-mistério com as marcas substituídas pelo caracter introduzido em todas as posições em que ele ocorra na palavra.
O jogo termina quando o utilizador acertar em todos os caracteres, ou quando atingir o máximo de jogadas sem sucesso.
Sugestão: use o predicado `random/1` com argumento `p` para escolher à sorte uma das `p` palavras-mistério que tem na sua base de conhecimento.

3.2 Master Mind

O desafio que se coloca desta vez é desenvolver um programa Prolog para jogar o jogo Master Mind com o utilizador, em que o computador detém a combinação de cores a adivinhar (4 entre 6 cores possíveis, sem repetições). O jogador terá de acertar na combinação em, no máximo, `n` jogadas.
O programa deve ser invocado através do predicado `masterMind/1`, em que o argumento único é o número de tentativas permitidas. Em cada jogada o computador lê uma combinação de 4 cores proposta pelo jogador e responde indicando o número de cores certas na posição certa (marcas pretas) e o número de cores certas na posição errada (marcas brancas).
O jogo termina quando o utilizador acertar na combinação (fizer 4 marcas pretas), ou quando atingir o máximo de jogadas sem sucesso.

[1] No xadrez uma Rainha põe em xeque todas as casa da mesma linha, coluna ou diagonais.
[2] No xadrez um Cavalo salta de casa onde para as 8 na sua vizinhança que se atinge deslocando uma em linha/coluna e uma para baixo/cima.

Sugestão: use o predicado `random/1` com argumento `c` para escolher à sorte uma das `c` combinações que tem na sua base de conhecimento.

3.3 Batalha Naval

Neste caso o que lhe é pedido é para recriar o tradicional jogo da Batalha Naval em Prolog. O computador deve carregar para a sua base de conhecimento uma configuração inicial do jogo, em que estarão colocados num tabuleiro de xadrez `10x10` os seguintes navios: 4 submarinos; 3 navios de 2 canos; 2 navios de 2 canos; 1 navio de 4 canos.
O programa deve ser invocado através do predicado `batalhaNaval/0`, em que o argumento único é o número de tentativas permitidas. Em cada jogada o computador lê 3 tiros (cada tiro é o par de coordenadas da casa que se pretende atingir, na forma de Linha e Coluna) e indica o resultado, isto é se foi tudo na água ou se algum navio foi atingido.
O jogo termina quando o utilizador afundar toda a frota, ou quando indicar que não quer continuar a jogar.

3.4 Minesweeper

Neste caso o programa que deve desenvolver em Prolog arranca, tal como no caso anterior com uma configuração inicial do jogo, em que estarão colocados num tabuleiro de xadrez `NxN` várias bombas (ao todo `M`).
O programa deve ser invocado através do predicado `minesweeper/2`, em que os argumentos são a dimensão do tabuleiro (`N`) e o número `M` de bombas enterradas no campo armadilhado. Em cada jogada o computador lê 1 par de coordenadas da casa a inspeccionar, na forma de Linha e Coluna, e indica o resultado, isto é mostra todas as casas em torno dessa que estão livres parando na fronteira, isto é, nas casas que estejam encostadas a uma bomba. Para cada quadrícula da fronteira, indica o número de bombas adjacentes.
O jogo termina quando o utilizador fizer explodir o campo, acertando num bomba, ou quando descobrir todas as quadrículas sem tocar em nenhuma bomba.