

Paradigmas da Programação MiECom (2º ano)

Trabalho Prático nº 2

Ano lectivo 2009/2010

1 Objectivos e Organização

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

- praticar o uso de *listas* e da *recursividade*;
- aumentar a experiência de programação na linguagem lógica Prolog para resolução geral de problemas;
- 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, como se pode ler em <http://caos.di.uminho.pt/~dpum/dpum> (ver mais especificamente CeNPL, em <http://paginas.fe.up.pt/~cenplf06/>, TIUP, em <http://caos.di.uminho.pt/~tiup/> MIUP, em <http://miup.estga.ua.pt/>).

Para o efeito, esta folha contém 8 enunciados, organizados em 2 tipos de jogos: interactivos (o utilizador joga contra o computador); e não-interactivos (jogos de tabuleiro, m que só o computador joga).

Para avaliação, cada grupo tem de resolver pelo menos um problema doo Grupo I e opcionalmente pode ainda resolver outro do Grupo II.

O programa desenvolvido—**a realizar até ao dia 20 de Dezembro**—deve ser enviado por email aos docentes por cada grupo, mal esteja concluído, acompanhado de um relatório de desenvolvimento.

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 I: jogos interactivos

2.1 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 character. Em cada jogada o computador lê uma letra e volta a mostrar a palavra-mistério com as marcas substituídas pelo character 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.

2.2 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.

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

2.3 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 $N \times N$ 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.

2.4 Bem e Mal

Ou o **Jogo da Verdade e da Mentira** em que, dada uma fórmula proposicional (proposta pelo jogador humano, ou gerada pelo computador) os dois jogadores têm objectivos contraditórios, um quer tornar a fórmula verdadeira e o outro quer torná-la falsa; uma jogada consiste em fixar o valor verdadeiro ou falso para uma variável proposicional da dita fórmula cujo valor ainda esteja em aberto; os jogadores vão jogando alternadamente, até que todas as variáveis tenham valor atribuído. Calculado o valor da fórmula (de acordo com os valores escolhidos para as variáveis e de acordo com as tabelas de verdade dos vários conectivos), ganha o jogador que atingiu o seu objectivo.

2.5 Galo Cervejeiro

Semelhante ao tradicional Jogo do Galo, jogador e computador defrontam-se sobre um tabuleiro quadrado (3x3) podendo em cada jogada: colocar numa casa vazia uma caneca de cerveja *cheia*, *meada*, *vazia*; ou beber um golo de cerveja de uma caneca não-vazia colocada numa casa do tabuleiro (cada golo é sempre exactamente igual a meia caneca). Ganha o primeiro que conseguir por em linha (horizontal, vertical, ou diagonal) três canecas com o mesmo nível (*cheia*, *meada*, *vazia*).

2.6 Quatro em Linha

É uma outra variante, bem conhecida, do Jogo do Galo no qual o objectivo é ir colocando peças numa das 7 colunas de um tabuleiro com 6 linhas, até se alinhar (na horizontal, vertical ou diagonal) 4 peças da mesma cor (do jogador ou do computador); como o tabuleiro está na vertical, o efeito da gravidade faz que cada peça colocada num coluna desça até à primeira casa livre a contar da base.

3 Grupo II: jogos não-interactivos

3.1 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 xeque¹.

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 conseqüentemente o número de Rainhas a colocar).

3.2 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 $N \times N$ a partir de uma casa dada, com saltos de cavalo² 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.

¹No xadrez uma Rainha põe em xeque todas as casa da mesma linha, coluna ou diagonais.

²No 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.