Universidade do Porto Faculdade de Engenharia FEUP	Faculdade de Engenharia da Universidade do Porto **Mestrado Integrado Engenharia Informática e Computação** **Mestrado Integrado Eng. Electrotécnica e de Computadores** **Programação em Lógica** *Época da Época Normal – Com Consulta / Duração: 3h00m*	**2009/2010** **3º Ano MIEIC/ 4º Ano MIEEC**
Nome:	**Data:**	**19/01/2010**

Nota: Utilize exclusivamente Prolog Standard no grupo I. No grupo II utilize Prolog Standard e a biblioteca CLPFD do SICStus Prolog. Responda em folhas separadas aos seguintes grupos de perguntas: Folha 1: Perguntas 1, 2, 3 e 4; Folha 2: Perguntas 5 e 6; Folha 3: Perguntas 7.1, 7.2 e 7.3.

GRUPO I – Programação em Prolog (13 val.)

1) Represente cada uma das seguintes frases em Prolog. (**1.5 Val**).

a) o joão apenas come chocolate, bananas ou queijo.
b) a raiz quadrada de 9 é 3 ou -3.
c) portugal, espanha e frança são todos países.
d) um português, espanhol, francês ou italiano é um latino.
e) com menos de 16 anos ou mais de 65 anos tem desconto no cinema.
f) os praticantes futebol, andebol ou ténis são desportistas.

2) Quais dos seguintes pares de termos unificam? Onde for relevante, indique as instanciações das variáveis que conduzem à unificação. (**1.5 Val**).

```
a) 'arroz' = arroz
b) 'Arroz' = arroz
c) Arroz = arroz
d) comida(arroz) = arroz
e) comida(arroz) = X
f) comida(X) = comida(arroz)
g) comida(arroz,X) = comida(Y, fiambre)
h) comida(arroz, X, cerveja) = comida(Y, fiambre, X)
i) comida(arroz, X, cerveja) = comida(Y, cheese_burger)
j) comida(X) = X
k) refeicao(comida(arroz), bebida(cerveja)) = refeicao(X, Y)
l) refeicao(comida(arroz), X) = refeicao(X, bebida(cerveja))
```

3) Como responde o Prolog às seguintes questões? Justifique. (**1.5 Val**).

```
a) ?- [a,b,c,d] = [a,[b,c,d]].
b) ?- [a,b,c,d] = [a|[b,c,d]].
c) ?- [a,b,c,d] = [a,b,[c,d]].
d) ?- [a,b,c,d] = [a,b,c|[d]].
e) ?- [a,b,c,d] = [a,b,c,d|[]].
f) ?- [] = _.
g) ?- [] = [_].
h) ?- [] = [_|[]].
```

4) Utilização do Cut (1.5 Val.).

Suponha o seguinte código Prolog.

```
a1(X,Y):- b(X), c(X,Y), d(Y).
a1(X,Y):- b(X), c(Y,Y), d(X).

a2(X,Y):- b(X), !, c(X,Y), d(Y).
a2(X,Y):- b(X), c(Y,Y), d(X).
```

```
b(1).
b(2).
c(1,4).
c(1,3).
```

```
c(2,4).
c(2,2).
d(1).
d(3).
d(4).
```

Indique justificando todas as respostas obtidas às seguintes questões. Apresente as respostas pela ordem pela qual são obtidas e justifique.

```
a) ?- a1(X,Y).
b) ?- a1(X,Y),!.
c) ?- a2(X,Y).
```

5) Tradução de Listas (3.0 Val.).

Suponha uma base de conhecimento com os seguintes factos:

```
traduz(1, one, um).
traduz(2, two, dois).
traduz(3, three, três).
traduz(4, four, quatro).
traduz(5, five, cinco).
traduz(6, six, seis).
traduz(7, seven, sete).
traduz(8, eight, oito).
traduz(9, nine, nove).
```

a) Escreva um predicado *traduz_lista(?Ingles,?Portugues)* que traduz um lista de números em Inglês para Português ou vice-versa (i.e. de Português para inglês). Nota: A lista original está toda na mesma língua. (**1.5 Val**).

Exemplo:

```
?- traduz_lista([one, nine, two], X).
X = [um, nove, dois].
?- traduz_lista (X, [um, sete, seis, dois]).
X = [one, seven, six, two].
```

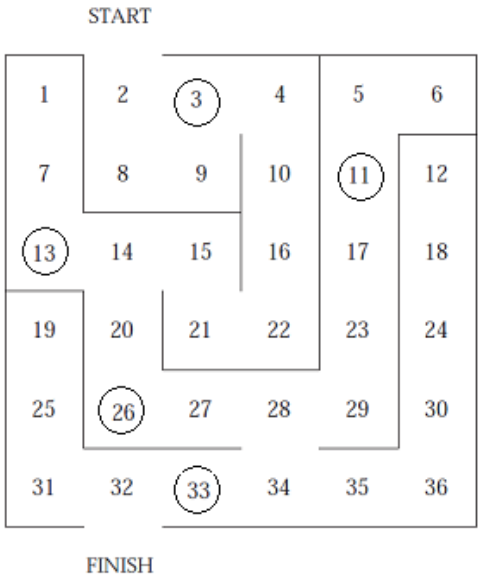
b) Escreva um predicado *ordena_lista(+Tipo,+Entrada, -Saida)* que ordena por ordem crescente ou decrescente um lista de números escrita em Inglês ou Português. (**1.5 Val**).

Exemplo:

```
?- ordena_lista(crescente, [one, nine, four, two], X).
X = [one, two, four, nine].
?- ordena_lista(decrescente, [um, sete, dois, seis, nove, dois], X).
X = [nove, sete, seis, dois, dois, um].
```

6) Labirinto (4.0 val.)

O objectivo deste problema consiste em encontrar um caminho através do labirinto desde o ponto de início (*start*) até ao ponto final (*finish*). O labirinto foi construído numa folha de papel quadriculado e tem a dimensão de 6x6, ou seja, 6 quadrados de largura por 6 de altura. Cada um dos quadrados está numerado conforme a figura. Considere as posições de inicio (start) e fim (finish) como posições do labirinto, perfazendo, assim, um total de 38 posições válidas. Se for possível efectuar um movimento de uma posição A para outra B existe um facto indicando-o que estas posições estão ligadas (*ligado(A,B).*). Um caminho no labirinto é uma lista de posições ligadas cujo primeiro elemento é a posição "*start*" e o último elemento é a posição "*finish*", de tal forma que cada posição na lista está ligada às posições anterior e seguinte. O programa não necessita de encontrar a solução óptima. De modo a não "andar em círculos", garanta que a próxima posição está (obviamente) ligada à actual mas não foi ainda visitada (i.e. não pertence à lista de posições que já foram visitadas).



a) Crie uma base de conhecimento contendo a informação do labirinto. (**0.5 Val**).

b) Crie um programa *resolve(-Solucao)* que partindo da posição "*start*"gere um caminho completo até "*finish*". (**2.0 Val**).

Exemplo:

```
?- resolve(Solucao).
Solucao=[start,2,3,4,10,16,22,21,15,14,20,26,27,28,34,33,32,finish].
```

c) Suponha a existência de ouro em determinadas quadrículas do tabuleiro, representada pelos factos ouro/1. No caso da figura, seria ouro(3). ouro(11). ouro(13). ouro(26). ouro(33).
Construa um predicado *resolve(-Solucao,-Num,-Casas)* que calcule, para além da solução do problema, o número de peças de ouro recolhidas e suas posições. **(1.5 Val).**

Exemplo:

```
?- resolve(Solucao, Num, Casas).
Solucao=[start,2,3,4,10,16,22,21,15,14,20,26,27,28,34,33,32,finish].
Num = 3.
Casas = [3,26,33].
```

GRUPO II – Programação em Lógica com Restrições (7.0 val)

7.1) Na seguinte multiplicação y x * 7 = z x x, cada letra x, y e z representa um dígito distinto. Pretende-se saber qual a soma dos valores x, y e z que tornam a equação anterior verdadeira? Utilizando programação em lógica com restrições, resolva o problema, calculando a soma das três variáveis. **(1.5 Val)**.

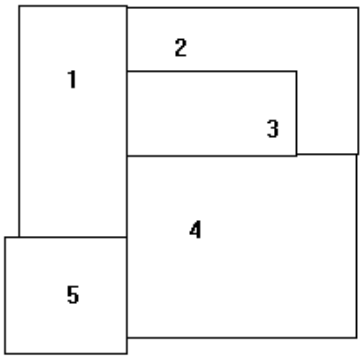
$$\begin{array}{r} y\ x \\ \times\ 7 \\ \hline z\ x\ x \end{array}$$

7.2) Pretende-se encontrar horários para aulas (com duração de 1 hora) leccionadas por diferentes docentes. Cada docente pretende leccionar num determinado intervalo. Todas as aulas são leccionadas na mesma sala (i.e. não pode haver aulas sobrepostas). Pretende-se ainda que as aulas leccionadas por mulheres sejam o mais cedo possível. Os dados são representados da seguinte forma:

```
docente(pedro, m, 3, 6).  % Pedro é do sexo masculino e pode leccionar
                          % aulas que comecem às 3, 4, 5 ou 6 horas
docente(joana, f, 3, 4).
docente(ana, f, 2, 5).
docente(joao m, 2, 4).
docente(david, m, 3, 4).
docente(maria, f, 1, 6).
```

Encontre uma solução genérica para este problema de optimização utilizando programação em lógica com restrições. (**2.5 Val**).

7.3) Suponha uma versão estendida do conhecido problema de colorir um conjunto de países num mapa com um dado número de cores, assumindo que países vizinhos são sempre coloridos com cores distintas. O programa deve incluir as seguintes restrições:



- Cardinalidade das cores. Cada cor é utilizada, no máximo, um dado número de vezes.
- Exclusão. Determinados países não podem ser pintados com determinadas cores.
- Condicionais. Se um país for pintado com uma cor então outro país tem/não pode ser pintado com uma dada cor.
- Diferentes. Embora não façam fronteira dois países têm de ser pintados com cores distintas.
- Preferências. Restrição flexível que indica que cada país tem preferência por um dado conjunto de cores. Deve-se maximizar o respeito pelas preferências.

Suponha que tem a informação representada num ficheiro com uma estrutura semelhante à indicada. Resolva este problema de optimização (versão genérica) utilizando programação em lógica com restrições. (**3.0 Val**).

```
paises(5).            % 5 países
cores(4,[2,1,1,2]).   %  4  cores.  Cores  1  e  4  são  usada  no  máximo  2  vezes
                        % e restantes cores são usadas, no máximo 1 vez.
fronteira(1,2).       % País 1 faz fronteira com o 2 (e o 2 faz fronteira com o 1).
fronteira(1,3).
fronteira(1,4).
fronteira(1,5).
fronteira(2,3).
fronteira(2,4).
fronteira(3,4).
fronteira(4,5).
exclui(1, [3,4]).     % País 1 não pode ser pintado com as cores3 e 4.
exclui(2, [3]).       % País 2 não pode ser pintado com a cor 3
seentao(1-4, 2-3).    % Se o país 1 tiver a cor 4, o país 2 tem a cor 3.
seexclui(1-3, 2-1).   % Se país 1 tiver a cor 3 então país 2 não pode ter a cor 1
diferentes(2,5).      %  Embora  não  façam  fronteira,  países  2  e  5  têm  cores
                      %  diferentes
preferências([1-[1,2], 2-[1], 3-[3,4], 4-[1,2]]).   % País 1 tem preferência pelas
% cores 1 e 2, país 2 tem preferência pela cor 1, país 3 tem preferência pelas
% cores 3 e 4 e país 4 tem preferências pelas cores 1 e 2.
```