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 – Proposta de resolução*		**2009/2010** **3º Ano MIEIC/** **4º Ano MIEEC**
Nome:		**Data:**	**19/01/2010**

GRUPO I – Programação em Prolog (13 val.)

1)

a)
```
come(joao,chocolate).
come(joao,bananas).
come(joao,queijo).
```

b)
```
raiz_quadrada(9,-3).
raiz_quadrada(9,3).
```

c)
```
pais(portugal).
pais (espanha).
pais (França).
```

d)
```
latino(X):- portugues(X).
latino (X):- espanhol(X).
latino (X):- frances(X).
latino (X):- italiano(X).
```

e)
```
desconto_cinema(X):- idade(X,Y), Y<16.
desconto_cinema(X):- idade(X,Y), Y>65.
```

f)
```
desportisa(X):- joga(X,futebol).
desportisa(X):- joga(X,andebol).
desportisa(X):- joga(X,tenis).
```

2)

```
a)yes
b)no
c)yes
d)no
e)yes
f)yes
g)yes
h)no
i)no
j)yes/no podem ambos estar correctos dependendo da justificação
k)yes
l)no
```

3)

```
a)no
b)yes
c)no
d)yes
e)yes
f)yes
g)no
h)no
```

4)

a)
```
X=1,Y=4;
X=1,Y=3;
X=2,Y=4;
x=1;Y=2;
```

```
b)
X=1,Y=4;

c)
X=1,Y=4;
X=1,Y=3;
```

5)

```
traduz(1, one, um).
traduz(2, two, dois).
traduz(3, three, tres).
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)
%traduz_lista(?Ingles, ?Port).
traduz_lista([], []).
traduz_lista([HI|RI], [HP|RP]):-
    traduz(_, HI, HP),
    traduz_lista(RI, RP).

b)
%ordena_lista(+Tipo, +Entrada, -Saida):-
ordena_lista(_, [], []).
ordena_lista(Tipo, L, SL) :- bubbleup(Tipo, L, ZL), !,
ordena_lista(Tipo, ZL, SL).
ordena_lista(_, SL, SL).

bubbleup(Tipo, [X, Y|L], [Y, X|L]) :- compara(Tipo, X, Y), !.
bubbleup(Tipo, [Z|L], [Z|ZL]) :- bubbleup(Tipo, L, ZL).

compara(Tipo,X,Y):-
    (traduz(N1, X,_); traduz(N1, _,X)),
    (traduz(N2, Y,_); traduz(N2, _,Y)),
    (Tipo==crescente, !, N1 > N2 ; N1 < N2).
```

6)

```
a)
ligado(start,2).
ligado(1,7). ligado(2,3). ligado(2,8). ligado(3,4). ligado(3,9).
ligado(4,10). ligado(5,6). ligado(5,11). ligado(7,13). ligado(8,9).
ligado(10,16). ligado(11,17). ligado(12,18). ligado(13,14). ligado(14,15).
ligado(14,20). ligado(15,21). ligado(16,22). ligado(17,23). ligado(18,24).
ligado(19,25). ligado(20,26). ligado(21,22). ligado(23,29). ligado(24,30).
ligado(25,31). ligado(26,27). ligado(27,28). ligado(28,29). ligado(28,34).
ligado(30,36). ligado(31,32). ligado(32,33). ligado(33,34). ligado(34,35).
ligado(35,36). ligado(32,finish).

b)
%resolve(-Solucao).

resolve(Solucao):- caminho(start, finish, [start], Solucao).

liga(X,Y):- ligado(X,Y).
liga(X,Y):- ligado(Y,X).

caminho(NoFim, NoFim, Lista, Lista).
caminho(NoInicio, NoFim, Lista, ListaFim):-
    liga(NoInicio, NoInterm),
    \+(member(NoInterm, Lista)),
    append(Lista, [NoInterm], Lista2),
    caminho(NoInterm, NoFim, Lista2, ListaFim).

%resolve(-Solucao, -Num, -Casas).

ouro(3). ouro(11). ouro(13). ouro(26). ouro(33).

resolve(Solucao, Num, Casas):-
    resolve(Solucao),
    findall(X, (ouro(X), member(X,Solucao)), Casas),
    length(Casas,Num).
```

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

7.1)

```
:-use_module(library(clpfd)).

puz1(Val):-
   Vars=[Y,X,Z],
   domain(Vars,0,9),
   Y#\=0, Z#\=0,
   all_distinct(Vars),
   (Y*10+X)*7 #= Z*100 +X*(11),!,
   labeling([],Vars),
   Val is Y+X+Z.
```

7.2)

```
solve(Vars):-
    findall(Sex-Ini-Fim, docente(_,Sex,Ini,Fim),Lista),
    length(Lista,NDoc),
    length(Vars,NDoc),
    dominios(Vars,Lista),
    all_distinct(Vars),
    avaliacao(Vars,Lista,FAval),!,
    minimize((labeling([],Vars),write(FAval-Vars),nl), FAval).

dominios([],[]).
dominios([H|RVars], [_-Ini-Fim|Resto]):-  H in Ini..Fim, dominios(RVars, Resto).

avaliacao([],[],0).
avaliacao([H|RVars],[Sex-_-_|Resto],FAval2):-
   (Sex==f, FAval2 #= FAval+H ; FAval2#=FAval),
   avaliacao(RVars,Resto,FAval).
```