1.
1.1
a)

```
cliente(1,'Joao','Porto').
cliente(2,'Joana','Porto').
cliente(3,'Mario','Coimbra').
cliente(4,'Maria','Leiria').
cliente(5,'Manuel','Lisboa').
```

b)

```
catalogo(100,'Fifia 2010','PC').
catalogo(101,'Fifia 2010','PS3').
catalogo(102,'Fifia 2010','XBOX').
catalogo(200,'Fifia 2011','PC').
catalogo(201,'Fifia 2011','PS3').
catalogo(202,'Fifia 2011','XBOX').
catalogo(300,'Bananoid 3D','PC').
catalogo(301,'Bananoid 3D','PS3').
catalogo(302,'Bananoid 3D','XBOX').
catalogo(400,'Assassins Script','PC').
catalogo(401,'Assassins Script','PS3').
catalogo(402,'Assassins Script','XBOX').
```

c)

```
stock(100,10).
stock(101,5).
stock(102,8).
stock(200,3).
stock(201,3).
stock(202,3).
stock(203,0).
stock(200,5).
stock(301,4).
stock(302,0).
stock(303,10).
stock(400,0).
stock(401,0).
stock(402,0).
```

1.2

```
a)cliente(Codigo,Nome,'Porto').
b)cliente(1,_,Morada).
c)catalogo(Codigo,_,'PC'),stock(Codigo,Quantidade),Quantidade>0.
```

1.3

```
stock:-findall(Jogo,(artigo(Codigo,Jogo,Plataforma),inventario(Codigo,Quantidade),
write(Jogo), write(' '),write(Plataforma),write(' '),write(Quantidade),nl),L).
```

1.4

```
validar_encomenda(Cliente,Jogo,Plataforma,Quantidade):-
cliente(_,Cliente,_),
catalogo(Codigo,Jogo,Plataforma),
stock(Codigo,Quantidade_em_stock),
Quantidade_em_stock>=Quantidade,!.
```

1.5

```
barras_vert(Lista_Existencias,Lista_Limites):-length(Lista_Existencias,N),
barras_vert(Lista_Existencias,Lista_Limites,N).

barras_vert(_,_,N):-N<0,!.
barras_vert(Lista_Existencias,Lista_Limites,N):-write(N),write(' '),
imprimeLinha(Lista_Existencias,Lista_Limites,N), nl,
N1 is N-1,
barras_vert(Lista_Existencias,Lista_Limites,N1).

imprimeLinha([],[],_):-!.
imprimeLinha([EE|RE],[EL|RL],N):-imprimeElemento(EE,EL,N),
imprimeLinha(RE,RL,N).

imprimeElemento(EE,EL,N):-N>EL,N=<EE,
write('X'), write(' '),!.
imprimeElemento(EE,EL,N):-N>EL,N>EE,
write(' '), write(' '),!.
imprimeElemento(EE,EL,N):-N=<EL,N>EE,
write('.'), write(' '),!.
imprimeElemento(EE,EL,N):-N=<EL,N=<EE,
write('O'), write(' '),!.
```

2.

a)estuda(joao,pl).
b)habitantes(franca,50).
c)pais_rico(alemanha).
d)autor(‘A Mensagem’,alguem).
e) mortal(X):- humano(X).
f) leva (jorge,guarda-chuva):- chover.
g) nao_ve_TV(X):- portar_mal(X).
h)seguranca(X):-altura(X,Y),mais_de(Y,1.8).

3.

a)[a,b|X]=[A,B,c] com A=a, B=b and X=[c].
b)falha
c) [a|[b,c]]=[a,b,c] com a=a, [b,c]=[b,c]
d)falha
e)falha
f)[a|[]]=[X] com X=a
g)falha
h)[H|T]=[[a,b],[c,d]] com H=[a,b] e T=[[c,d]].
i)falha
j)falha

4.

a)

```
extrai_pos_par([],[]).
extrai_pos_par([_],[]).
extrai_pos_par([_,X|Xs],[X|Ys]):- extrai_pos_par(Xs,Ys).
```

b)varia de acordo com a solução apresentada por cada um na alínea anterior

c)

```
extrai_geral([],[],_):-!.
extrai_geral([X|Xs],[X|Ys],Predicado):-
F=..[Predicado,X],
F,!,
extrai_geral(Xs,Ys,Predicado).
extrai_geral([_|Xs],Resultado,Predicado):- extrai_geral(Xs,Resultado,Predicado).
```

5.

Nota: esta solução só dá uma solução, que alterações faria para dar as (eventuais) restantes soluções por backtracking?

```
domino(Chain) :-
findall(pedra(X,Y),pedra(X,Y),[FirstStone|RestStones]),
Chain = [FirstStone|RestChain],
init_freesides(FirstStone,FreeSides),
chain(FreeSides,RestStones,RestChain),!.
init_freesides(pedra(A,A),FreeSides) :- !, FreeSides = [A,A,A].
init_freesides(pedra(A,B),FreeSides) :- FreeSides = [A,B].
chain( _, [], []).
chain(FreeSides,Stones,[Stone|Chain]) :-
my_select(Stone,Stones,RestStones),
add_stone(Stone,FreeSides,RestStones,Chain).
add_stone(pedra(A,A),FreeSides,RestStones,Chain) :- !,
my_select(A,FreeSides,RestFreeSides),!,
chain([A,A|RestFreeSides],RestStones,Chain).
add_stone(pedra(A,B),FreeSides,RestStones,Chain) :-
(
my_select(A,FreeSides,RestFreeSides),!,
chain([B|RestFreeSides],RestStones,Chain)
;
my_select(B,FreeSides,RestFreeSides),!,
chain([A|RestFreeSides],RestStones,Chain)
).

my_select(X, [X|Tail], Tail).
my_select(Elem, [Head|Tail], [Head|Rest]) :-
        my_select(Elem, Tail, Rest).
```