UNIVALI – CTTMar Kobrasol – Ciência da Computação – Paradigmas de Programação 4ºper
Profa: Fernanda Cunha Exercícios M2 2018/1 – parte A

Exercício 1. Calculo de Fatorial e Fibonacci

a) Construa um predicado para calcular o fatorial de um número N: `fatorial(N,Valor).`

b) Construir um predicado para calcular o fibonacci de um número N.

Série de Fibonacci : N 0 1 2 3 4 5 6 ...
Fib. 1 1 2 3 5 8 13

Exercício 2. Funcionamento das Listas [H|T] – teórico

Preveja os resultados das seguintes questões em Prolog:

```
a) ?- [a|[b,c,d]] = [a,b,c,d].
b) ?- [H|T] = [a, b, c, d, e].
c) ?- [H|T] = [a, [b,c,d]].
d) ?- [H|T] = [].
e) ?- [One, Two | T] = [apple, sprouts, fridge, milk].
```

Exercício 3. Funcionamento das Listas [H|T] – teórico

Resolva as igualdades, dizendo quais são os valores finais das variáveis.

```
a) ?- lista([a,[b],c,[d]]) = lista([_|[X|X]]).
b) ?- lista([[a],[b],C])=lista([C,B,[a]]).
c) ?- lista([c,c,c])=lista([X|[X|_]]).
d) ?- lista([a,[b,c]])=lista([A,B,C]).
e) ?- [1,2,3,4,5,6,7]=[X,Y,Z|D].
```

Exercício 4. Listas de Números.

a) Construa o predicado `lista_ate(N,L)` que devolva a lista L de todos os números inteiros entre 1 e N.
b) Construa o predicado `lista_entre(N1,N2,L)` que devolva a lista L de todos os números inteiros entre N1 e N2 (ambos incluídos).
c) Construa o predicado `soma_lista(L,Soma)` que some todos os elementos da lista L, obtendo como resultado Soma.
d) Escreva o predicado `par(N)` que dado um número inteiro N, determine se ele é ou não um número par.
e) Escreva o predicado `lista_pares(N,Lista)` que aceite um número inteiro e que determine a lista de todos os números pares iguais ou inferiores a esse número.
f) Construa o predicado `ultimoElemento(Lista,X)` para encontrar o último elemento de uma lista qualquer.
g) Construa o predicado `maiorElemento(Lista,X)` para encontrar o maior elemento de uma lista qualquer.
h) Construa o predicado `media(L,X)`, onde X é o valor médio dos valores contidos na lista L.
i) Construa o predicado `escore(X,Y,A,B)` onde X e Y são listas de inteiros do mesmo tamanho, A é o número de posições que possuem números idênticos e B é o número de elementos que ocorrem simultaneamente em ambas as listas, mas em posições diferentes.
   Ex.: ?-`escore([7,2,3,4], [2,3,2,4], A, B).` Resp.: A=1 (o valor 4) e B=2 (os valores 2 e 3).
   Dica: o predicado inicial deve chamar outros 2, cada um resolvendo um dos itens de resposta.
j) Construa o predicado `palindromo(X)` que é verdadeiro se X é uma lista cujos elementos invertidos produzem a mesma ordem original.

Exercício 5. Efeito do Cut

Suponha a seguinte base de fatos em Prolog

```
dados(um).
dados(dois).
dados(tres).
```

a) Qual o resultado da seguinte pergunta?

```
cut_teste_a(X) :- dados(X).
cut_teste_a('ultima_clausula').
?- cut_teste_a(X), write(X), nl, fail.
```

b) Qual o resultado do seguinte programa com um Cut no final da primeira clausula?

```
cut_teste_b(X):- dados(X), !.
cut_teste_b('ultima_clausula').
?- cut_teste_b(X), write(X), nl, fail.
```

c) Qual o resultado do seguinte programa com um Cut no meio dos dois objetivos?

```
cut_teste_c(X,Y) :- dados(X), !, dados(Y).
cut_teste_c('ultima_clausula').
?- cut_teste_c(X,Y), write(X-Y), nl, fail.
```

Exercício 6. Cuts Verdes e Vermelhos

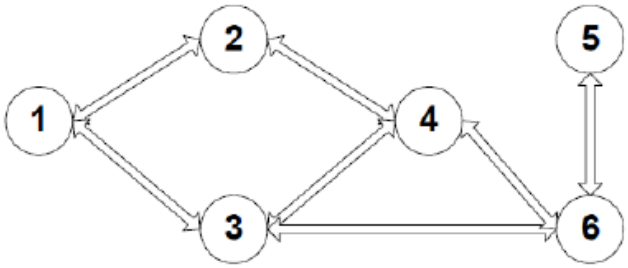
Explique a função dos 'cuts' incluídos no programa abaixo.

```
imaturo(X):- adulto(X), !, fail.
imaturo(X).
adulto(X):- pessoa(X), !, idade(X, N), N>=18.
adulto(X):- tartaruga(X), !, idade(X, N), N>=50.
```

Exercício 7. Pesquisa de Ligação com Visita a uma Lista de Nós

Considere que um grafo não dirigido é representado por um conjunto de cláusulas unitárias da forma ***ligacao(No1, No2)***. **Exemplo:**

ligacao(1, 2).
ligacao(1, 3).
ligacao(2, 4).
ligacao(3, 4).
ligacao(3,6).
ligacao(4, 6).
ligacao(5,6).



a) Escreva um predicado ***caminho(+NoInicio, +NoFim, -Lista),*** que dados dois nós do grafo, calcule um possível caminho (não necessariamente o mais curto) entre esses nós. *Nota: Suponha que a dimensão máxima do caminho é 5.* **Exemplos:**

```
?- caminho(2, 3, Lista).
```

Lista = [2,4,3] ; Lista = [2,4,6,3] ; Lista = [2,1,3] ; false

```
?- caminho(1, 5, Lista)  % Solução possível: de nó 1 para 2, depois para 4, para 6 e finalmente para nó 5.
```

Lista = [1,2,4,6,5] ; Lista = [1,2,4,3,6,5] ; Lista = [1,3,4,6,5] ; Lista = [1,3,6,5] ; false

```
?- caminho(2, 2, Lista).
```

Lista = [2,4,2] ; Lista = [2,4,6,3,1,2] ; Lista = [2,4,3,1,2] ; Lista = [2,1,2] ; Lista = [2,1,3,4,2] ; Lista = [2,1,3,6,4,2] ; false

b) Escreva um predicado ***ciclos(+No, +Comp, -Lista),*** que dado um nó, calcule todos os ciclos possíveis, com comprimento inferior a ***Comp***, desse nó. *Sugestão: Utilize o predicado caminho (alínea anterior) como base para a resolução.* ***Exemplo:***

?- ciclos(4, 3, Lista).
Lista = [[4,3,6], [4,6,3]]

?- ciclos(4, 5, Lista).
Lista = [[4,3,6], [4,6,3], [4,2,1,3], [4,3,1,2]]