Lista de Exercícios - Paradigma Lógico - Prolog

1. Construa um banco de dados para representar os membros de uma família. Crie o predicado progenitor(X, Y) para representar o relacionamento X é progenitor de Y. Represente em Prolog as relações sexo (masculino ou feminino), irmã, irmão, descendente, mãe, pai, avô, tio e primo. Os predicados progenitor e sexo devem ser fatos no banco de dados, os demais devem ser descritos como regras.

2. Escreva, sem cortes, um predicado separa que separa uma lista de inteiros em duas listas: uma contendo os números positivos (e zero) e outra contendo os números negativos. Por exemplo:

```
?- separa([3, 6, -2, 5, -1], P, N).
P = [3, 6, 5],
N = [-2, -1].
```

Melhore este programa com a ajuda do corte, sem alterar o seu significado.

3. Escreva um predicado segundo(Lista, X) que verifica se X é o segundo elemento de Lista.

4. Escreva um predicado iguaisL(Lista1, Lista2) que verifica se Lista1 é igual a Lista2.

5. Escreva um predicado duplica(E, S) em que o primeiro argumento é uma lista e o segundo argumento é uma lista obtida a partir da primeira duplicando cada um dos seus elementos. Por exemplo:

```
?- duplica([3, a, j7], S).
S = [3, 3, a, a, j7, j7].
```

6. Escreva um predicado ultimo(L, X) que é satisfeito quando o termo X é o último elemento da lista L.

7. Escreva um predicado soma(L, R) que é satisfeito quando o termo R é igual a soma dos elementos da lista L. Assuma que a lista L possua apenas valores numéricos.

8. Conserte o predicado fatorial(N, F) para que não entre em loop em chamadas onde N é um número negativo e nem em chamadas verificadoras, onde ambos N e F vêm instanciados.

```
fatorial(0, 1) :- !.
fatorial(N, F) :- M is N - 1, fatorial(M, F1), F is F1*N.
```

9. Defina um predicado contiguo(L), que testa se uma lista tem dois elementos contíguos iguais. Exemplo:

```
?- contiguo([a,s,s,d,f,e]).
true
?- contiguo([a,s,d,f,e]).
false
```

10. Defina um predicado particiona(L, L1, L2), que recebe uma lista L de elementos como primeiro argumento e retorna duas listas como resposta. A lista L1 é formada pelos elementos de L que estejam em posições ímpares. A lista L2 é formada pelos elementos de L que estejam em posições pares. Exemplo:

```
?- particiona([1,2,4,6,2], L1, L2).
L1 = [1,4,2], L2 = [2,6]
```

11. Defina um predicado mdc(A, B, R) que retorne o máximo divisor comum entre dois números. Exemplo:

```
?- mdc(12, 18, R).
R = 6
```

12. Escreva um predicado insereOrdenado que tenha três argumentos. O primeiro argumento é uma lista ordenada chamada List (assuma ordem crescente), o segundo é um elemento X e o terceiro argumento é a lista obtida com a inserção de X em List, de modo que esta ainda esteja ordenada. Exemplo de uso:

```
?- insereOrdenado([2,6,9], 5, R).
R = [2,5,6,9]
```

13. Faça um predicado que indique o enésimo termo de uma lista qualquer. O predicado deve possuir três argumentos: o primeiro argumento é um número que indica a posição do enésimo termo da lista (assuma que o primeiro termo assuma a posição de índice 1 da lista); o segundo argumento é a lista propriamente dita; e o terceiro argumento é o valor do enésimo termo. Exemplo de uso:

```
?- enésimo(2,[f,w,q,a,h],R).
R = w
```

14. Escreva um predicado remover(L1,X,L2) que é satisfeito quando L2 é a lista obtida pela remoção da primeira ocorrência de X em L1. Exemplo de uso:

```
?- remover([a,s,f,e,s,w], s, L2).
L2 = [a, f, e, s, w]
```

15. Escreva um predicado remover_todas(L1,X,L2) que é satisfeito quando L2 é a lista obtida pela remoção de qualquer ocorrência de X em L1. Exemplo de uso:

```
?- remover_todas([a,s,f,s,e,s,w], s, L2).
L2 = [a, f, e, w]
```

16. Escreva um predicado remover_repetidos(L1,L2) que é satisfeito quando L2 é a lista obtida pela remoção dos elementos repetidos em L1. Exemplo de uso:

```
?- remover_reptidos([a,s,a,f,s,f,s,f,a,e,s,w], L2).
L2 = [a, s, f, e, w]
```

17. Escreva um predicado conjunto(L1, L2) que é satisfeito quando L1 é uma lista formada apenas por elementos presentes em L2. Assuma que o predicado será utilizado apenas com todos os argumentos instanciados. Exemplo de uso:

```
?- conjunto([1,2,3,3,1,1,4,32,4,4,2],[1,2,3,4,32]). true. ?-
conjunto([1,2,3,3,1,1,4,32,4,4,2],[1,2,3,4]). false.
```

18. Escreva um predicado produto(A, B, P) que é satisfeito quando P é igual ao produto de todos números compreendidos no intervalo [A,B]. Exemplo de uso:

```
?- produto(3,5,P).
P = 60
```

19. Escreva um predicado inverter(L1,L2) que é satisfeito quando L2 possui seus elementos em ordem inversa a L1. Exemplo de uso:

```
?- inverter([4,5,2,3,1,7],L2).
L2 = [7,1,3,2,5,4]
```

20. Faça um programa PROLOG que com o estado de um tabuleiro do jogo da velha diga se há vencedor e quem é este vencedor (cruz ou bola). Exemplo de uso:

```
?- velha([[x,x,o],[x,o,x],[o,x,o]],X).
X = o.
?- velha([[x,x,o],[x,o,x],[x,o,o]],X).
X = x.
?- velha([[x,o,x],[o,o,x],[o,x,o]],X).
false.
```

21. Considere a seguinte base de fatos em Prolog:

```
aluno(joao, calculo).
aluno(maria, calculo).
aluno(joel, programacao).
aluno(joel, estrutura).
frequenta(joao, puc).
frequenta(maria, puc).
frequenta(joel, ufrj).
professor(carlos, calculo).
professor(ana_paula, estrutura).
professor(pedro, programacao).
funcionario(pedro, ufrj).
funcionario(ana_paula, puc).
funcionario(carlos, puc).
```

Escreva as seguintes regras em Prolog: a) Quem são os alunos do professor X? b) Quem são as pessoas que estão associadas a uma universidade X? (alunos e professores)

22. Suponha os seguintes fatos:

```
nota(joao,5.0).
nota(maria,6.0).
nota(joana,8.0).
nota(mariana,9.0).
nota(cleuza,8.5).
nota(jose,6.5).
nota(jaoquim,4.5).
nota(mara,4.0).
nota(mary,10.0).
```

Considerando que:
Nota de 7.0 a 10.0 = Aprovado
Nota de 5.0 a 6.9 = Recuperação
Nota de 0.0 a 4.9 = Reprovado
Escreva uma regra para identificar a situação de um determinado aluno.