Primeira Prova de Técnicas Alternat. de Prog. (05/2.000)
Nome: ———————— Assinatura: ————————-

Questão 1 (1,0 ponto)

Para cada frase da lingua Portuguesa dada abaixo, escreva, em Lógica de Predicados, uma expressão correspondente com "sentido" aproximado.

(a) Todo programa em Prolog é compacto

(b) Todo aluno estudioso

(c) Alguns feriados são chuvosos

(d) Quem sabe faz e quem não sabe ensina

Questão 2 (1,0 ponto)

Diga, com poucas palavras, o que entende pelos conceitos de Unificação e Backtracking.

Questão 3 (2,5 pontos)

Construir um predicado em Prolog, denominado "invertida", o qual relaciona uma lista de itens (possivelmente vazia) com uma outra lista a qual contém exatamente todos os itens da primeira, porém em ordem invertida. Seu comportamento é o expresso abaixo:

```
?- invertida([a], X).
X = [a] ?
yes

?- invertida([1,2,3], X).
X = [3, 2, 1] ?
yes

?- invertida([a,b,c,d,e,f], X).
X = [f, e, d, c, b, a] ?
yes
```

OBS: Caso queira, assuma a existência do predicado "concatena" (o qual relaciona 3 listas, onde a terceira e' a junção direta das 2 primeiras) para ajudar na definição do predicado "invertida". Exemplo: concatena([1,2], [3,4], [1,2,3,4]).

Questão 4 (2,5 pontos)

Construir um predicado em Prolog, denominado "simetrica", o qual relaciona uma lista de itens (possivelmente vazia) com seu próprio valor verdade, resultando em "verdadeiro" quando a lista e' simétrica e "falso" quando a lista nao é simétrica. Seu comportamento é o expresso abaixo:

```
?- simetrica([ a, b ]).
no

?- simetrica([ a, a ]).
yes

?- simetrica([ a, b, b, a ]).
yes

?- simetrica([ a, b, f, b, a ]).
yes

?- simetrica([ a, b, c, d, e, f ]).
no
```

OBS: Caso queira, assuma a existência do predicado "invertida" (o qual relaciona duas listas por meio da ordem inversa de seus elementos) para ajudar na definição do predicado "simetrica" (exemplo acima).

Questão 5 (3,0 pontos)

É dado o programa Prolog abaixo:

```
pertence_a(X, [X | Cauda]).
pertence_a(X, [Cabeca | Cauda]) :-
    pertence_a(X, Cauda).

elem_repetidos([], []).
elem_repetidos([X | Cauda1], [X | Cauda2]) :-
    elem_repetidos(Cauda1, Cauda2),
    pertence_a(X, Cauda1),
    not(pertence_a(X, Cauda2)).
elem_repetidos([X | Cauda1], L2) :-
    elem_repetidos(Cauda1, L2).
```

Ativando-se o TRACE interativo para o predicado “elem_repetidos”, foi obtida o seguinte sequência de execuo para a pegunta abaixo.

```
?- elem_repetidos([a,a,b], Z).
** (1) Call : elem_repetidos([a, a, b], _1)?
** (2) Call : elem_repetidos([a, b], _2)?
** (3) Call : elem_repetidos([b], _3)?
** (4) Call : elem_repetidos([], _4)?
** (4) Exit : elem_repetidos([], [])?
** (5) Call : pertence_a(b, [])?
** (5) Fail : pertence_a(b, [])?
** (4) Redo : elem_repetidos([], [])?
** (4) Fail : elem_repetidos([], _4)?
** (6) Call : elem_repetidos([], _3)?
** (6) Exit : elem_repetidos([], [])?
** (3) Exit : elem_repetidos([b], [])?
** (7) Call : pertence_a(a, [b])?
** (8) Call : pertence_a(a, [])?
** (8) Fail : pertence_a(a, [])?
** (7) Fail : pertence_a(a, [b])?
** (3) Redo : elem_repetidos([b], [])?
** (6) Redo : elem_repetidos([], [])?
** (6) Fail : elem_repetidos([], _3)?
** (3) Fail : elem_repetidos([b], _3)?
** (9) Call : elem_repetidos([b], _2)?
** (10) Call : elem_repetidos([], _5)?
** (10) Exit : elem_repetidos([], [])?
** (11) Call : pertence_a(b, [])?
** (11) Fail : pertence_a(b, [])?
** (10) Redo : elem_repetidos([], [])?
** (10) Fail : elem_repetidos([], _5)?
** (12) Call : elem_repetidos([], _2)?
** (12) Exit : elem_repetidos([], [])?
** (9) Exit : elem_repetidos([b], [])?
** (2) Exit : elem_repetidos([a, b], [])?
** (13) Call : pertence_a(a, [a, b])?
** (13) Exit : pertence_a(a, [a, b])?
** (14) Call : pertence_a(a, [])?
** (14) Fail : pertence_a(a, [])?
** (1) Exit : elem_repetidos([a, a, b], [a])?
Z = [a] ?
yes
```

Pede-se uma forma alternativa de definição do predicado elem_repetidos de maneira a obtermos uma redução do número de passos necessrios para se alcancar as mesmas soluções. Explique sua decisão.

DICAS: Tal redução poderia ser atingida com a diminuição da necessidade de Backtracking (expresso acima pela presença do Redo nas linhas do TRACE interativo. Outra forma é por meio da redução de chamadas diretas(Call).