Capítulo 7

Prolog (Listas 2)

7.6 Listas

Os exercícios desta aula devem ser resolvidos usando predicados de ordem superior, ou seja, os predicados que definir não devem usar recursão.

1. Defina o predicado `insere_ordenado/3`, tal que `insere_ordenado(El, Lst1, Lst2)`, em que `El` é um inteiro e `Lst1` e `Lst2` são listas de inteiros ordenadas de forma ascendente, significa que `Lst2` é o resultado de inserir `El` em `Lst1`. Por exemplo,

```
?- insere_ordenado(5, [1,2,3,6,9], L).
L = [1, 2, 3, 5, 6, 9] .

?- insere_ordenado(0, [1,2,3,6,9], L).
L = [0, 1, 2, 3, 6, 9] .

?- insere_ordenado(10, [1,2,3,6,9], L).
L = [1, 2, 3, 6, 9, 10] .

?- insere_ordenado(3, [1,2,3,6,9], L).
L = [1, 2, 3, 3, 6, 9] .
```

2. Defina o predicado `junta_novo_aleatorio/4`, tal que `junta_novo_aleatorio(Lst1, Lim_Inf, Lim_Sup, Lst2)`, em que `Lst1` e `Lst2` são listas de inteiros ordenadas de forma ascendente, e `Lim_Inf` e `Lim_Sup` são dois inteiros, tais que `Lim_Inf` $\leq$ `Lim_Sup`, significa que `Lst2` é o resultado de inserir em `Lst1` um inteiro aleatório `Al`, tal que `Lim_Inf` $\leq$ `Al` $\leq$ `Lim_Sup`, e `Al` não pertence a `Lst1`. Por exemplo,

```
?- junta_novo_aleatorio([1,3], 1, 5, Lst2).
Lst2 = [1, 3, 5] .

?-  junta_novo_aleatorio([1,3], 1, 5, Lst2).
Lst2 = [1, 2, 3] .

?-  junta_novo_aleatorio([1, 2, 3, 4, 5], 1, 5, Lst2).
false.
```

Para gerar um inteiro aleatório entre dois limites, use o predicado `random_between(Lim_Inf, Lim_Sup, N)`. Sugestão: use o predicado `insere_ordenado(El, Lst1, Lst2)` definido anteriormente, e o predicado pré-definido `subtract(Lst1, Lst2, Lst3)`.

3. Defina o predicado `repete_el/3`, tal que `repete_el(El,N,L)` significa que `L` é a lista constituída por `N` (um inteiro não negativo) ocorrências do elemento `El`. Sugestão: use o predicado `length(Lst, Comp)`. Por exemplo,

```
?- repete_el(a,3,L).
L = [a, a, a]

?- repete_el(a, 0, L).
L = [] .
```

4. Defina o predicado `duplica_elementos/2`, tal que `duplica_elementos(Lst1, Lst2)` significa que a lista `Lst2` é o resultado de repetir cada um dos elementos da lista `Lst1`. Por exemplo,

```
?- duplica_elementos([a,b,c], L).
L = [a, a, b, b, c, c].

?- duplica_elementos([], L).
L = [].
```

5. Defina o predicado `num_occ/3`, tal que `num_occ(Lst, El, N)` significa que o elemento `El` ocorre `N` vezes na lista `Lst`. Por exemplo,

```
?- Lst = [1,2,3,4,1,5,1], num_occ(Lst, 1, N).
Lst = [1, 2, 3, 4, 1, 5, 1],
```

```
N = 3.

?- Lst = [1,2,3,4,1,5,1], num_occ(Lst, 11, N).
Lst = [1, 2, 3, 4, 1, 5, 1],
N = 0.
```

6. Defina o predicado `substitui_maiores_N/4`, tal que `substitui_maiores_N(N, Subst, Lst1, Lst2)`, em que `Lst1` é uma lista de inteiros e `N` é um inteiro, significa que a lista `Lst2` é o resultado de substituir todos os elementos da lista `Lst1` maiores do que `N`, por `Subst`. Por exemplo,

```
?- substitui_maiores_N(5, maior_5, [1,5,6,3,7], L) .
L = [1, 5, maior_5, 3, maior_5] .

?- substitui_maiores_N(5, maior_5, [], L) .
L = [] .
```