Árvores e processamento simultâneo - Prática

Marco A L Barbosa malbarbo.pro.br

1. Projete uma função que receba como entrada uma árvore binária `t` e um número `n` e devolva uma nova árvore binária que é como `t` mas com `n` somado a cada elemento.
2. Projete uma função que verifique se uma árvore binária é uma árvore binária de busca. Uma árvore binária de busca tem as seguintes propriedades: 1) A subárvore a esquerda contém valores nos nós menores que o valor no nó raiz. 2) A subárvore a direita contém valores nos nós maiores que o valor no nó raiz. 3) As subárvores a esquerda e a direita também são árvores binárias de busca.
3. Projete uma função que verifique se um elemento está em uma árvore binária de busca.
4. Projete uma função que receba como entrada uma lista aninhada `lst` e devolva uma nova lista aninhada como os mesmo elementos de `lst`, mas em ordem reversa.

   ```
   > (reverse* (list (list 2 3) 8 (list 9 (list 10 11) 50) (list 10) 70))
   '(70 (10) (50 (11 10) 9) 8 (3 2))
   ```

5. Utilizando apenas as funções primitivas `zero?`, `add1` e `sub1`, escreva as funções `>`, `>=`, `<`, `<=` e `=`. Cada função deve receber como parâmetro dois números naturais e executar a operação de comparação apropriada.
6. Defina uma função que receba como entrada duas listas de números em ordem crescente, e devolva uma nova lista com os elementos das duas listas de entrada em ordem crescente.

   ```
   > (intercala (list 3 7 12) (list 2 4 5))
   '(2 3 4 5 7 12)
   ```

7. Projete uma função que devolve os n primeiros elementos de uma lista.

   ```
   > (mantem (list 10 40 70 20 3) 2)
   '(10 40)
   ```

8. Projete uma função que devolve um nova lista sem os n primeiros elementos de uma dada lista.

   ```
   > (descarta (list 10 40 70 20 3) 2)
   '(70 20 3)
   ```

9. Projete uma função que devolva uma nova lista que é como a lista de entrada mas sem o elemento de uma dada posição.

   ```
   > (remove-em (list 3 6 1 2) 2)
   '(3 6 2)
   ```

10. Projete uma função que devolva uma lista que é como a lista de entrada mas com um dado elemento inserido em uma dada posição.

    ```
    > (insere-em (list 3 6 1 2) 5 2)
    '(3 6 5 1 2)
    ```

11. Projete uma função que devolva a sublista com um determinado intervalo de uma lista.

    ```
    > (sublista (list 10 20 30 40 50) 1 4)
    '(20 30 40)
    ```

12. Projete uma função que devolva uma nova lista que é como a lista de entrada mas com os elementos rotacionados n posições a esquerda.

```
> (rotaciona-esquerda (list 10 20 30 40 50) 2)
'(30 40 50 10 20)
```