LP2 - Linguagem Funcional

Segundo Exame Escrito

Sérgio Soares
04 de Abril de 2002

1. Complete a defição do ADT abaixo que deve representar uma árvore binária de busca de inteiros onde a função `emptyTree` inicializa uma árvore vazia (1.0). Defina as funções `addElement` (1.0) que insere um elemento na árvore, e `isInTree` (1.0) que verifica se um elemento está na árvore. Defina ainda funções para somar, `sumTree` (1.0), e dobrar, `doubleTree` (1.0) os elementos da árvore.

```
module Tree (Tree, emptyTree, ...)
  where
  data Tree = ...

  emptyTree :: Tree
  emptyTree = NULL

  ...
```

2. (1.0) Considerando o ADT definido na questão anterior escreva a expressão que cria a seguinte árvore:

```
          4
        /   \
      2       6
    /  \    /  \
  1     3  5     7
                   \
                     8
```

3. (3.0) Considerando o ADT definido na questão 1 e suas funes prove que `sumTree (doubleTree x) = 2 * sumTr` para todas as árvores x finitas.

4. (1.0) O que devemos acrescentar na prova acima para provar que a propriedade é válida para todas as árvores incluindo as infinitas? Por que? Complete então a prova para todas as árvores incluindo as infinitas.