Assignment 9

Design an algorithm, isPermutation(A,B) that takes two sequences A and B and determines whether or not they are permutations of each other, i.e., they contain the same elements but possibly occurring in a different order. Assume the elements in A and B cannot be sorted, and that A and B cannot be modified. Hint: A and B may contain duplicates. (Same problem as in previous homework, but this time use a dictionary to solve the problem.)

```
Algorithm isPermutation(A, B):
    if A.size() ≠ B.size() then
        return False

    D ← new empty Dictionary(HashTable)

    for each element a in A.elements() do
        if D.findElement(a) ≠ null then
            D.replaceElement(a, D.findElement(a) + 1)
        else
            D.insertElement(a, 1)

    // Subtract elements from B
    for each element b in B.elements() do
        if D.findElement(b) = null then
            return False
        D.replaceElement(b, D.findElement(b) - 1)
        if D.findElement(b) < 0 then
            return False

    return True
```

Design a pseudo code algorithm isBalanced(T) that decides whether or not a binary tree, T, is a balanced binary tree. For this problem, we define “balanced” to mean that the height of the left and right sub-trees of every node do not differ by more than one. For example, if the left sub-tree has height 5, then the right sub-tree can only have height 4, 5, or 6. Similarly, if the right sub-tree has height 4, then the left sub-tree must have height 3, 4, or 5.

```
Algorithm isBalanced(T):
    (balanced, height) ← checkBalance(T, T.root())
    return balanced



Algorithm checkBalance(T, node):
    if node = null then
        return (True, -1)

    (leftBalanced, leftHeight) ← checkBalance(T.leftChild(node))
    if not leftBalanced then
        return (False, 0)

    (rightBalanced, rightHeight) ← checkBalance(T.rightChild(node))
    if not rightBalanced then
        return (False, 0)

    if Math.abs(leftHeight - rightHeight) > 1 then
        return (False, 0)

    height ← 1 + max(leftHeight, rightHeight)
    return (True, height)
```