QUEENS COLLEGE Department of Computer Science
CSCI 313 Practice problems on Search Trees

Instructor: Alex Ryba

These are some of the solutions to practice problems. Not all problems have solutions here.

Solutions to some older problems might not make use of generics. Generics are now required in this course.

Problem 1 For each of the following structures list the additional requirements that a binary tree must satisfy to qualify.

(a) A binary search tree.

Binary Search Tree Order: Inorder traversal shows the data values in the tree in increasing order.

(b) A binary heap.

1. Heap Order: Every node stores a smaller data value than the values in any of its children.

2. Heap Shape: All levels of the tree except the last are completely full and the last level is partially filled from the left.

(c) An AVL Tree.

1. It is a binary search tree.

2. It satisfies AVL balance: For every node the left and right heights differ by at most 1.

(d). Implement the following method:

```
BNode<T> leftmostRightDescendant(BNode<T> n)
```

The method should return the leftmost right descendant of a binary node (or return *null* if there is no right descendant.

```
public BNode<T> leftmostRightDescendant(BNode<T> n) {
   try {
      BNode<T> m = n.getRight();
      while (m.getLeft() != null) m = m.getLeft();
      return m;
   } catch (Exception e) {
      return null;
   }
}
```

Problem 2 Consider the following diagram showing the state of a binary tree.

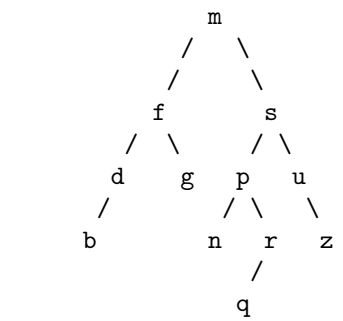

Diagram T.

(a) Write down the inorder traversal of the tree.

bdfgmnpqrsuz

(b) Write down the postorder traversal of the tree.

bdgfnqrpzusm

(c) List all single lower case letters whose insertion into an AVL Tree represented by Diagram T would require a rebalance of the tree. (Remember: AVL trees can not contain duplicate data items.)

acvwxy

(d) Assume that Diagram T represents the state of an AVL Tree. Show how the tree is changed when the data item g is removed.

It is modified through the following sequence of changes