CSCI 305, Homework # 8

YOUR NAME HERE

Due date: Midnight, Tuesday, June 5

Binary search trees with equal keys. This is problem 12-1 in the book.

Equal keys pose a problem for the implementation of binary search trees.

a. What is the asymptotic performance of TREE-INSERT when used to insert n items with identical keys into an initially empty binary search tree?

Sould just be n as the tree grows down the left or right side. Each insert takes N actions at worst thus the total to insert N things is $O(N^2)$.

We propose to improve TREE-INSERT by testing before line 5 to determine whether $z.key == x.key$ and by testing before line 11 to determine whether $z.key == y.key$.

If equality holds, we implement one of the following strategies. For each strategy, find the asymptotic performance of inserting n items with identical keys into an initially empty binary search tree. (The strategies are described for line 5, in which we compare the keys of z and x. Substitute y for x to arrive at the strategies for line 11.)

b. Keep a boolean flag $x.b$ at node x, and set x to either $x.left$ or $x.right$ based on the value of $x.b$, which alternates between FALSE and TRUE each time we visit x while inserting a node with the same key as x.

The flipping back and forth will make the tree have a height of lg n. We do n inserts so the performance is $O(n\ lg\ n)$.

c. Keep a list of nodes with equal keys at x, and insert z into the list.

All inserts will go to the same place, and the tree height is not effected so it is a constant amount of time. Thus n inserts will be $O(n)$.

d. Randomly set x to either $x.left$ or $x.right$. (Give the worst-case performance and informally derive the expected running time.)

In the worst case on left or right will be selectd and we will end up with teh same tree from part a. where all nodes get inserted down one side. This leads to the same $O(n^2)$

In the expected case we get a 50/50 distribution so we end up wit hthe same tree from part b. this gives $O(n\ lg\ n)$.