CS 61B Homework 7
Due noon Wednesday, April 9, 2014

This homework will give you practice with 2-3-4 trees. This is an individual assignment; you may not share code with other students.

Copy the Homework 7 directory by doing the following, starting from your home directory.

```
    cp -r ~cs61b/hw/hw7 .
```

Your task is to implement the insert operation for a 2-3-4 tree, in the file Tree234.java (in the dict package). For simplicity, the tree stores only keys; no element is associated with each key. Furthermore, the keys are ints.

The central two operations have the following prototypes.

```
  public boolean find(int key);
  public void insert(int key);
```

find() returns true if the specified key is in the tree, and false otherwise. find() is already implemented for you; inspecting the code will help you understand the data structure better. insert() inserts its parameter "key" into the tree. Please implement the top-down insertion algorithm described in the Lecture 27 notes, and not the bottom-up insertion algorithm described by Goodrich and Tamassia.

Because there is no object associated with each key, there is no reason to store duplicate keys. Hence, if a key is found to already be in the tree, don't insert another copy.

A toString() method is also provided for 2-3-4 trees, and is useful for testing. DO NOT CHANGE IT; the autograder will use it to check your trees.

There's also a printTree() method, which prints a 2-3-4 tree in an easier-to-read, but more space-consuming, tree-shaped manner (albeit sideways). This method will not be tested, and you may change it to your liking.

The Tree234Node class represents a node in a 2-3-4 tree, and has the following fields.

```
  int keys;
  int key1;
  int key2;
  int key3;
  Tree234Node parent;
  Tree234Node child1;
  Tree234Node child2;
  Tree234Node child3;
  Tree234Node child4;
```

The "keys" field is the number of keys stored in the node, and must be 1, 2, or 3. The fields "key1", "key2", and "key3" are filled in (in order) with the int keys stored in the node. If keys == 1, the value of key2 doesn't matter. If keys <= 2, the value of key3 doesn't matter.

The "parent" field is the node's parent. The fields "child1" through "child4" are the node's children, and are filled in in order from child1 to child4. All four of these references must be set to null in a leaf node. If keys == 1, child3 should be null, and if keys <= 2, the value of child4 should be null.

You may not change any of these invariants regarding how a Tree234Node is represented, but you may add helper methods to the Tree234 and Tree234Node classes, and you may add new fields to the Tree234 class (but not the Tree234Node class) if it helps. However, please make sure that the toString() methods and the find() method work correctly in their unmodified forms. DO NOT CHANGE toString() or find().

Test code is provided in the main method of the Tree234 class. To compile, change (cd) to your hw7 directory and type "javac -g dict/*.java". To run, type "java dict.Tree234". The test code does not test whether you correctly update the tree's "size" field; you'll have to test that yourself. The test code doesn't test all cases; consider adding more test code if time permits.

Submitting your solution

Change (cd) to your hw7 directory, which should contain the dict directory. The dict directory should contain Tree234.java and Tree234Node.java. You're not allowed to change IntDictionary.java, so you can't submit it. Make sure your homework compiles and runs on the _lab_ machines just before you submit.

From your hw7 directory, type "submit hw7". (Note that "submit" will not work if you are inside the dict directory!) After submitting, if you realize your solution is flawed, you may fix it and submit again. You may submit as often as you like. Only the last version you submit before the deadline will be graded.