CS140 Final Exam - Fall, 2005 - Answers

Question 1: (6 points)

- Tree A: Not an AVL tree, because "Edgerrin" is imablanced (LH child has a height of 2, and RH child has a height of zero).
- Tree B: Not an AVL tree, because "Dallas" is less than "Edgerrin," but is the right child of "Edgerrin"
- Tree C: Not an AVL tree, because "Marvin" is not binary.
- Tree D: Is an AVL tree.

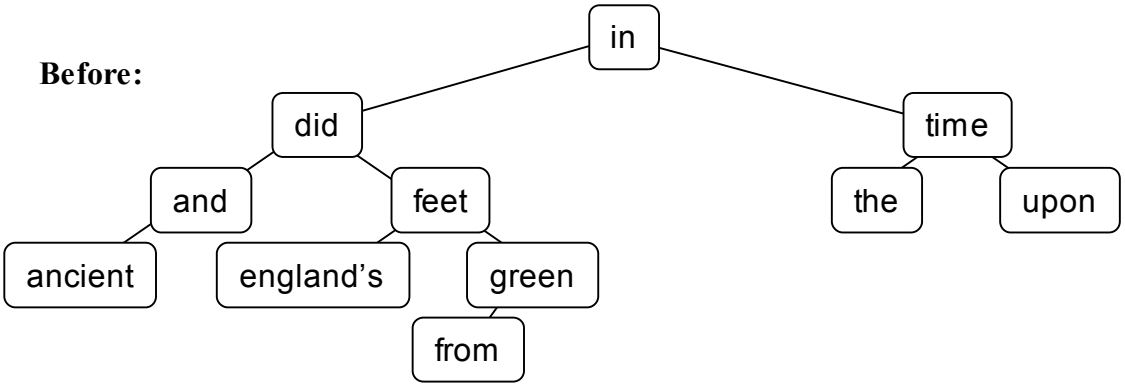
Question 2: (3 points)

PREORDER	**POSTORDER**
Marvin	Dallas
Brandon	Brandon
Dallas	Edgerrin
Edgerrin	Reggie
Peyton	Peyton
Reggie	Marvin

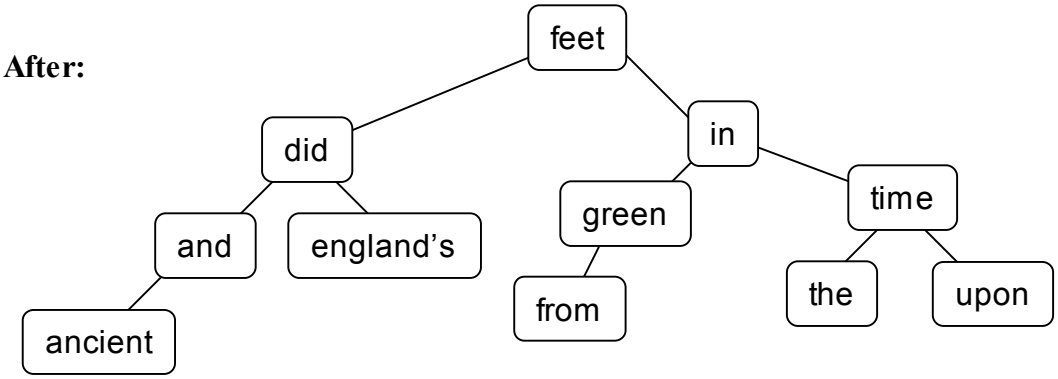
Question 3: (4 points)

When you insert "from", it will be green's left child, and the root node is imbalanced. To balance it requires a double-rotation about "feet". Here are the before and after pictures -- the "after picture is the answer.

Before:

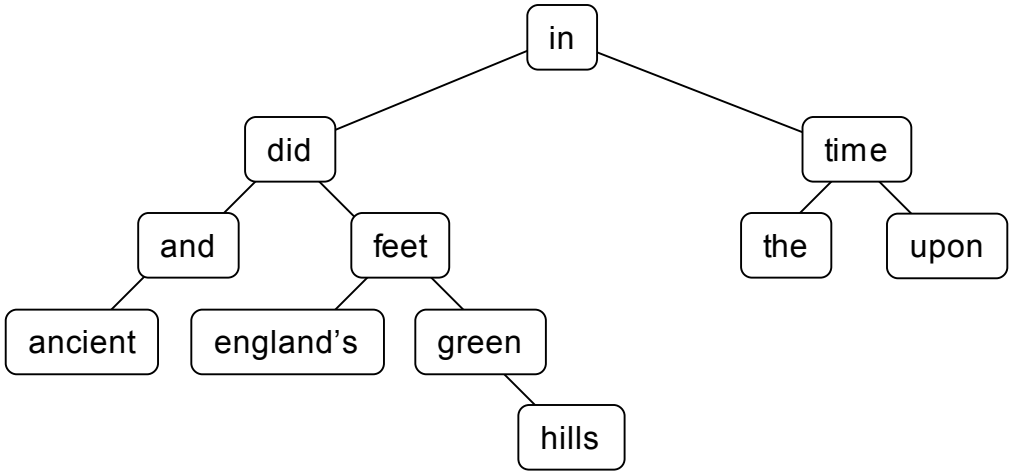


After:

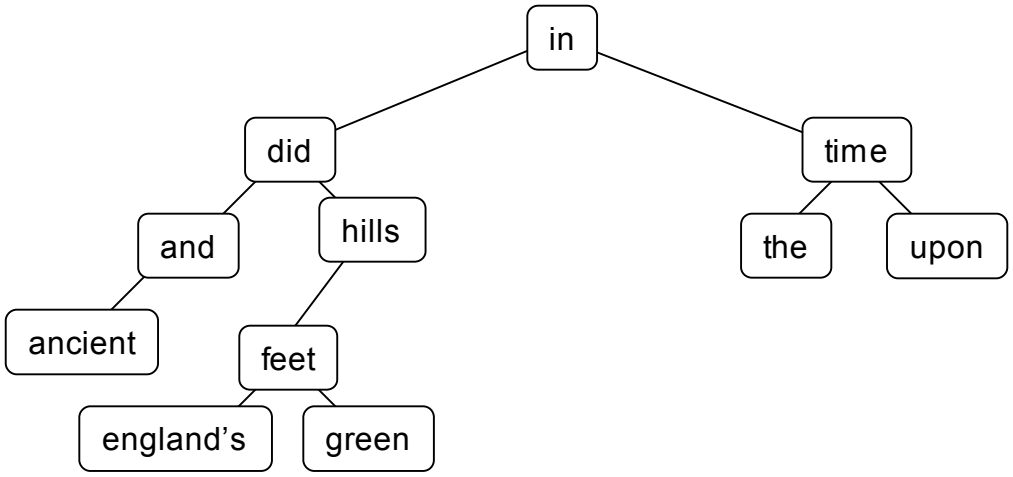


CS140 Final Exam - Fall, 2005 - Answers

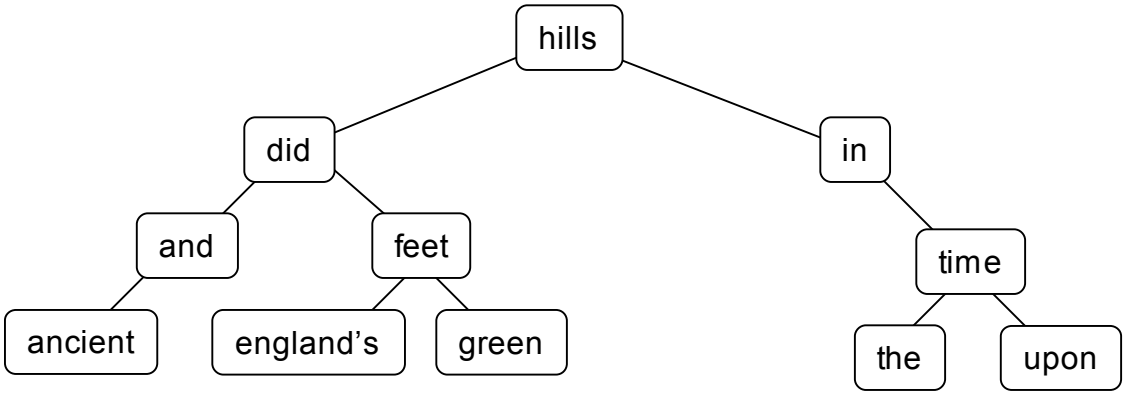
Question 4: (4 points) First you put "hills" as the right child of "green":



Then you perform a zig-zig rotation about "Hills":



Then a zig-zag rotation to yield the final tree:



CS140 Final Exam - Fall, 2005 - Answers

Question 5: (3 points) Each insertion can be *O(n)*, where there are n nodes in the tree. However, if *M* is large enough, then *M* insertions into the tree will be $MO(\log n)$.

Question 6: (4 points)

1. It should be fast.
2. It should randomize its input data. Put another way, the range of hash values should be distributed uniformly from the given input keys.

Question 7: (8 points)

This is a recursive traversal. For each node in the tree, print out the node, prepended by "level" spaces, then recursively call print_tree on the tree rooted at the node's val field:

```
print_tree(JRB t, int level)
{
  JRB tmp;
  int i;

  jrb_traverse(tmp, t) {
    for (i = 0; i < level; i++) printf(" ");
    printf("%s\n", tmp->key.s);
    print_tree((JRB)tmp->val.v, level+1);
  }
}
```

Grading:

Q1 - 1.5 points per part. 0.75 correctness, 0.75 reason on the incorrect ones.
Q2 - 1.5 points per part - no partial credit.
Q3 - 2 points for giving a valid AVL tree where "feet" is the root.
2 more points for giving the correct AVL tree.
Q4 - 1 point for giving a valid binary search tree (containing all elements) where "hills" is the root. 1.5 more points for getting the correct tree, with perhaps the first rotation wrong. 2 more points for giving the correct tree.
Q5 - 1.5 points for saying each insertion can be *O(n)*.
1.5 points for saying that multiple insertions average out to $O(\log n)$ - that's a less precise way of saying what I said.
Q6 - 2 points for fast. 2 points for spreading out the hash values uniformly. You had to say the word "uniformly" or convey that fact to get your two points.
Q7 - 2 points for a decent for loop.
2 points for printing out the spaces and the key.
2 points for a plausible recursive call.
2 points for getting the details right.