1. Problem 4-6 (VLSI chip testing) - This is a really good divide and conquer problem I left out of the last hw

2. Exercise 12.2-4 (Prof. Bunyan’s property)

3. Exercise 12.4-2 (“Describe a binary search tree on \( n \) nodes such that the average depth...”)

4. Problem 12-3 (Average Node Depth in Randomly Built Binary Search Tree)

5. Exercise 13.1-6 (Largest number of nodes with black-height \( k \))

6. Exercise 13.3-1 (“In line 16 of RB-Insert ...”)

7. Problem 13-3 (AVL Trees)

8. Problem 13-4 (Treaps)

9. HAFTs: A half-full tree (HAFT) is a rooted binary tree that is a useful data structure for designing self-healing networks. Let \( \ell \) be a positive integer. For \( \ell \) a power of 2, the complete tree with \( \ell \) leaf nodes is the unique haft with \( \ell \) leaf nodes. For \( \ell \) not a power of 2, a tree with \( \ell \) leaf nodes is a haft if and only if (1) the root node, \( r \), has two children; (2) the left subtree of \( r \) is the root of a complete binary containing \( 2^\left\lfloor \log \ell \right\rfloor \) leaf nodes; and (3) the right subtree of \( r \) is a haft. (Recall that a complete binary tree is one where every internal node has two children and every leaf node has the same depth)

Show the following by induction:

- For all positive \( \ell \), there is a unique haft with \( \ell \) leaf nodes.
• Call the haft with \( \ell \) leaf nodes \( haft(\ell) \). Then, the height of \( haft(\ell) \) is \( \lceil \log n \rceil \).

10. **Challenge:** In the self-healing application of hafts, the leaf nodes are associated with actual machines in a network, and the internal nodes represent additional “router nodes” (a scarce resource). To merge a list of hafts, \( h_1, h_2, \ldots, h_x \) we want to create a single new haft, \( h \), which contains as leaf nodes all the leaf nodes in \( h_1, h_2, \ldots, h_x \), and adds the smallest number of new internal nodes as possible.

• Show how you can merge a collection of \( x \) hafts, each of size no more than \( n \), into a single big haft by adding no more than \( O(x \log n) \) internal nodes.

Hint: Think about how to set up a correspondence between binary numbers and hafts, and binary addition and haft merging.