1. Given the keys 1, 4, 5, draw a binary search tree with...
   (a) 1 at the root.
   (b) 4 at the root.
   (c) 5 at the root.

   There may be multiple trees possible. You need only give one.

2. Given the keys 1, 3, 4, 5, 9, 10, 16, show all possible binary search trees...
   (a) with 5 at the root and with a tree height of 2 (meaning, there are 2 edges from the root to the
       furthest leaf).
   (b) with 4 at the root, 1 and 16 as leaves, and with a tree height of 3.
   (c) with 1 at the root, 3 and 16 as leaves, and with a tree height of 5.

3. Consider the binary-search-tree property and the min-heap property.
   (a) What is the difference between the two?
   (b) Can the min-heap property be used to print out the keys of an n-node tree in sorted order in
       O(n) time? Show how, or explain why not.

4. Give a nonrecursive algorithm that performs an inorder tree walk, printing the keys in order. Assume
   you have a stack to work with.

5. Suppose that we have numbers between 1 and 1000 in a binary search tree and we want to search for
   the number 363. Indicate whether each of the following sequences could or could not be the sequence
   of nodes examined. Hint: draw out the path and see if the path obeys the binary tree property.
   (a) 2, 252, 401, 398, 330, 344, 397, 363
   (b) 924, 220, 911, 244, 898, 258, 362, 363
   (c) 925, 202, 911, 240, 912, 245, 363
   (d) 2, 399, 387, 219, 266, 382, 381, 278, 363
   (e) 935, 278, 347, 621, 299, 392, 358, 363

6. Consider the complete binary search tree of height 3 on the keys 1, 2, ..., 15. In your drawings, don’t
   worry about the NIL nodes.
   (a) Draw the tree and assign each row a red or black color such that the tree has a black-height of 4.
       Indicate if this is not possible and why.
   (b) Draw the tree and assign each row a red or black color such that the tree has a black-height of 2.
       Indicate if this is not possible and why.
   (c) Draw the tree and assign each row a red or black color such that the tree has a black-height of 1.
       Indicate if this is not possible and why.

7. Consider the tree in figure 13.1. Suppose you add a node n with key 36.
   (a) Who is the n’s parent?
   (b) If n is red, is the resulting tree a red-black tree? If not, which property is violated?
   (c) If n is black, is the resulting tree a red-black tree? If not, which property is violated?

8. Suppose that we “absorb” every red node in a red-black tree into its black parent, so that the children
   of the red node become children of the black parent. How many children might a parent now have?
   Assume every internal node of the original tree has two children. The resulting tree after the absorb
   may not be a binary tree.
9. Show that the right rotate operation of a red-black tree retains binary search tree properties. You do not need to refer to the algorithm, just to figure 13.2 on page 313. You will show this using inequalities between $x, y, \alpha, \beta,$ and $\gamma$.

10. Argue that in every $n$-node binary search tree, there are exactly $n - 1$ possible rotations.