Tutorial (week 7)

All the answers can be given with pseudocode or with Python. Remember that most of the time, many solutions are possible.

1. Suppose you are given the array $A = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]$, and you then perform a binary search algorithm to find the number 8. Which numbers in the array A are compared against the number 8 ?

2. Insert items with the following keys (in the given order) into an initially empty binary search tree: 30, 40, 50, 24, 8, 58, 48, 26, 11, 13. Draw the tree that results.

3. From the final tree of the previous question, add the element 32 and then remove the elements 8 and 40. Note that two answers are possible, depending on how you select the node to reinsert.

4. Suppose you have a binary search tree, T, storing numbers in the range from 1 to 500, and you do a search for the integer 250. Which of the following sequences are possible sequences of numbers that were encountered in this search. For the ones that are possible, draw the search path, and, for the ones that are impossible, say why.

   (a) (2, 276, 264, 270, 250)

   (b) (100, 285, 156, 203, 275, 250)

   (c) (475, 360, 248, 249, 251, 250)

   (d) (450, 262, 248, 249, 270, 250)

5. Draw the binary search trees of minimum and maximum heights that store all the integers in the range from 1 to 7, inclusive.

6. Give a pseudocode description of an algorithm to find the element with smallest key in a binary search tree. What is the running time of your method ?

7. A certain Professor Amongus claims that the order in which a fixed set of elements is inserted into a binary search tree does not matter (the same tree results every time). Give a small example that proves Professor Amongus wrong.

8. Suppose that a binary search tree, T, is constructed by inserting the integers from 1 to n in this order. Give a big-Oh characterization of the number of comparisons that were done to construct T.

9. Suppose you are given a sorted array, A, of n distinct integers in the range from 0 to n, so there is exactly one integer in this range missing from A. Describe an $O(\log n)$-time algorithm for finding the integer in this range that is not in A.

10. Describe how to perform the operation `findAllElements`(k), which returns every element with a key equal to k (allowing for duplicates) in an ordered set of n key value pairs stored in an ordered array, and show that it runs in time $O(\log n + s)$, where s is the number of elements returned.

11. Describe how to perform the operation `findAllElements`(k), as defined in the previous exercise, in an ordered set of key-value pairs implemented with a binary search tree T, and show that it runs in time $O(h + s)$, where h is the height of T and s is the number of items returned.

Hints

- **Question 1:** be sure to keep the low, mid, and high variables straight during the algorithm simulation.
- **Question 2:** Review the algorithm for inserting items in a binary search tree.
- **Question 3:** Review the definition of a binary search tree.
- **Question 4:** Review the definition of a binary search tree.
- **Question 5:** Review the selection algorithm given in the book, and specialize it for the case $i = 1$.
- **Question 7:** Recall the worst-case example of the height of a binary search tree.
- **Question 8:** Think about how to devise a probe test to determine if you are querying above or below the missing number.