Data Structures and Algorithms – (COMP SCI 2C03)
Winter 2021
Tutorial-5

March 1, 2021

1. Give five orderings of the keys A X C S E R H that, when inserted into an initially empty BST, produce the best-case tree.
   Answer: Any sequence that inserts H first; C before A and E; S before R and X.

2. Suppose that a certain BST has keys that are integers between 1 and 10, and we search for 5. Which sequence below cannot be the sequence of keys examined?

   a. 10, 9, 8, 7, 6, 5

   b. 4, 10, 8, 7, 5

   c. 1, 10, 2, 9, 3, 8, 4, 7, 6, 5

   d. <u>2, 7, 3, 8, 4, 5</u>

   e. 1, 2, 10, 4, 8, 5

3. Consider the BST given in Figure 1. Draw the resulting tree when

   a. The key H is inserted into it.

   b. Delete the key T from the tree obtained after (a).

   c. Give the sequence of nodes visited to compute the minimum and maximum operations, on the tree obtained from (b).

   d. Give the sequence of nodes visited to compute the floor(C) and ceiling(F) operations, on the tree obtained from (b).

   e. What is the `rank` of the node Q in the tree obtained from (b).

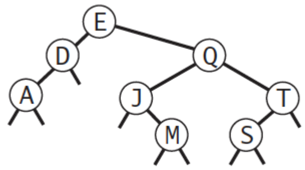

Figure 1: Question 3

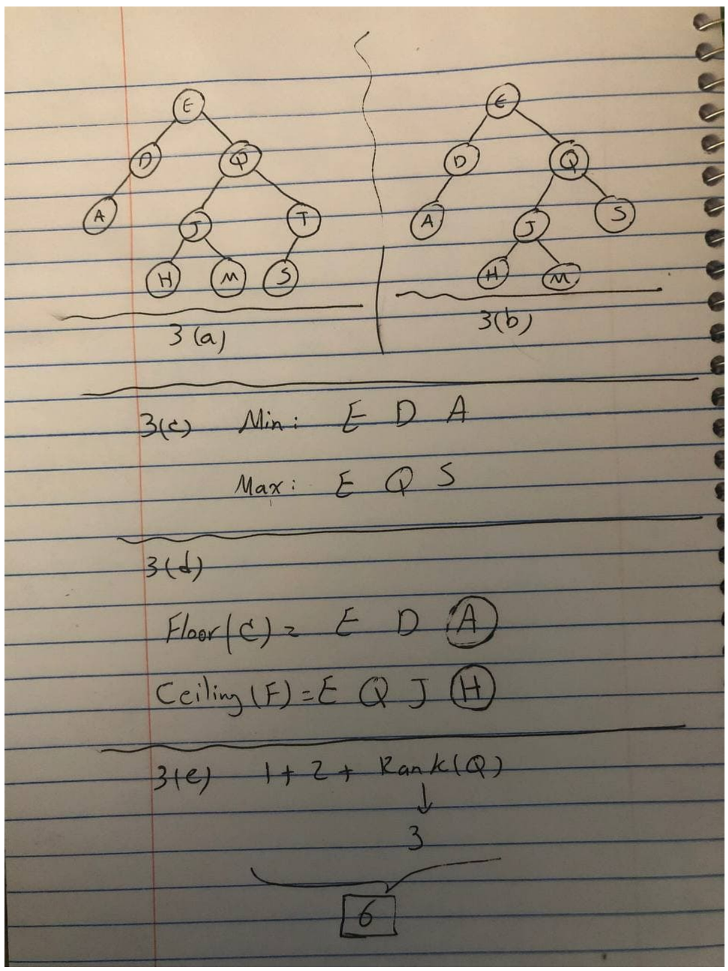

Figure 2: Question 3 (Answer)

4. Draw the (i) 2-3 tree and (ii) red-black tree that results when you insert the keys Y L P M X H C R A E S in that order into an initially empty

tree. **Answer:** (i) Figure 3 and (ii) Figure 4

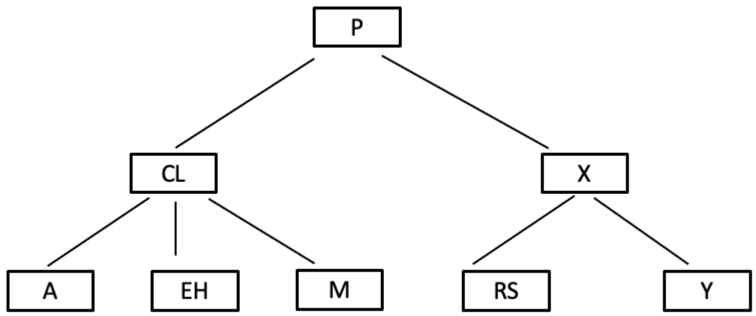

Figure 3: Solution for Q4(i)

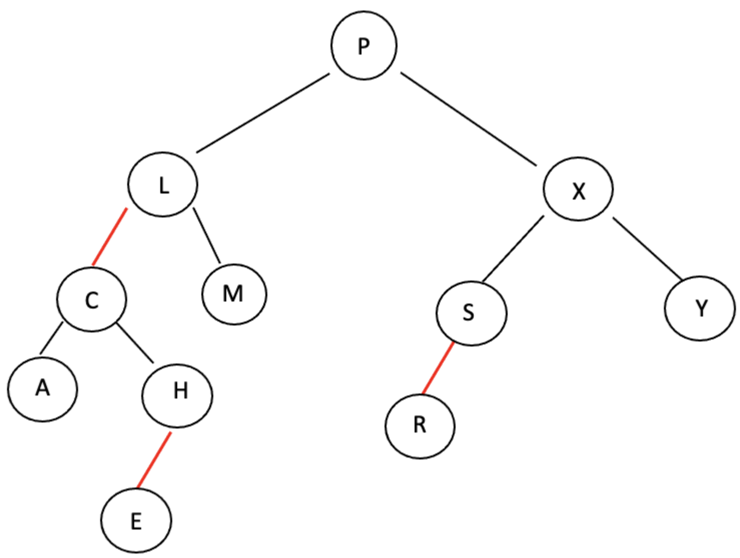

Figure 4: Solution for Q4(ii)

5. Find an insertion order for the keys S E A R C H X M that leads to a 2-3 tree of height 1.
   Answer: The insertion order for keys to get a 2-3 tree of height 1 is: A R X (E H) (C M S) (Any ordering of the triple (A R X) will work, and any ordering of the pair E and H will work. Also any ordering of the triple C M S will work.)

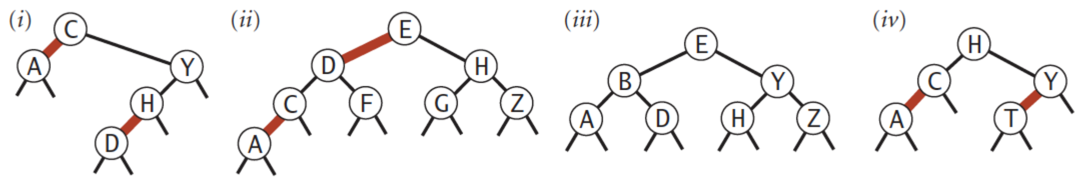

Figure 5: Question 6

6. Which of the trees given in Figure 5 are red-black BSTs?

   Answer: The trees given in (iii) and (iv) are RBTs. The tree given in (i) is not an RBT as it does not have perfect black balance, and the tree given in (ii) is not an RBT as it does not have perfect black balance nor symmetric order.

7. Draw the red-black BST that results when you insert letters A through K in order into an initially empty tree, then describe what happens in general when trees are built by insertion of keys in ascending order.
   Answer: see `https://algs4.cs.princeton.edu/33balanced/` for the animation. whether you add keys in increasing order or decreasing order to an initially empty RBT, the height of the RBT is balanced after each insert operation, and is always $O(logN)$. However, note that, when keys are added in an increasing order the height of the RBT increases monotonically. The detailed construction is provided in Figure 6 and Figure 7. As you see, when we insert the items in order, the red-black BST balances its height in $O(log(n))$ time after each insertion. Note that, this order of insertion will create the worst case scenario for the normal BST.

8. Draw the red-black BST that results when you insert letters A through K in reverse order into an initially empty tree, then describe what happens in general when trees are built by insertion of keys in descending order.
   Answer: see `https://algs4.cs.princeton.edu/33balanced/` for the animation. The detailed construction is provided in Figure 8 and Figure 9. If we add keys in decreasing order to an initially empty RBT, the height of the RBT is balanced after each insert operation, and is

always $O(logN)$. Similar to Question 7, we see a worst case scenario of insertion for a normal BST, where the red-black BST balances its height in $O(log(n))$ time after each insertion.

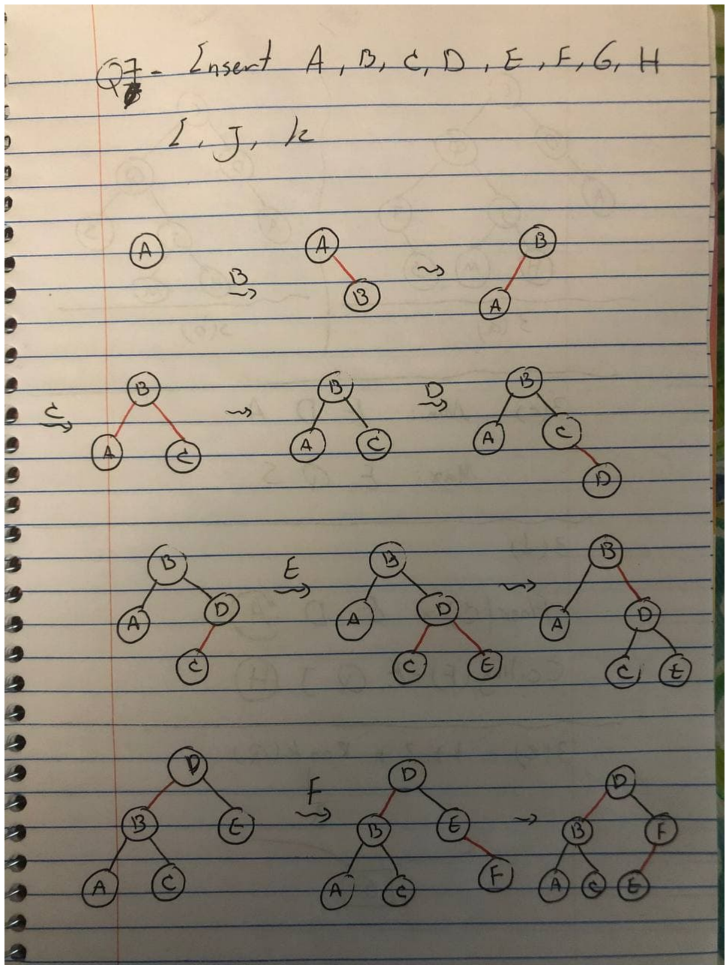

Figure 6: Question 7 (a)

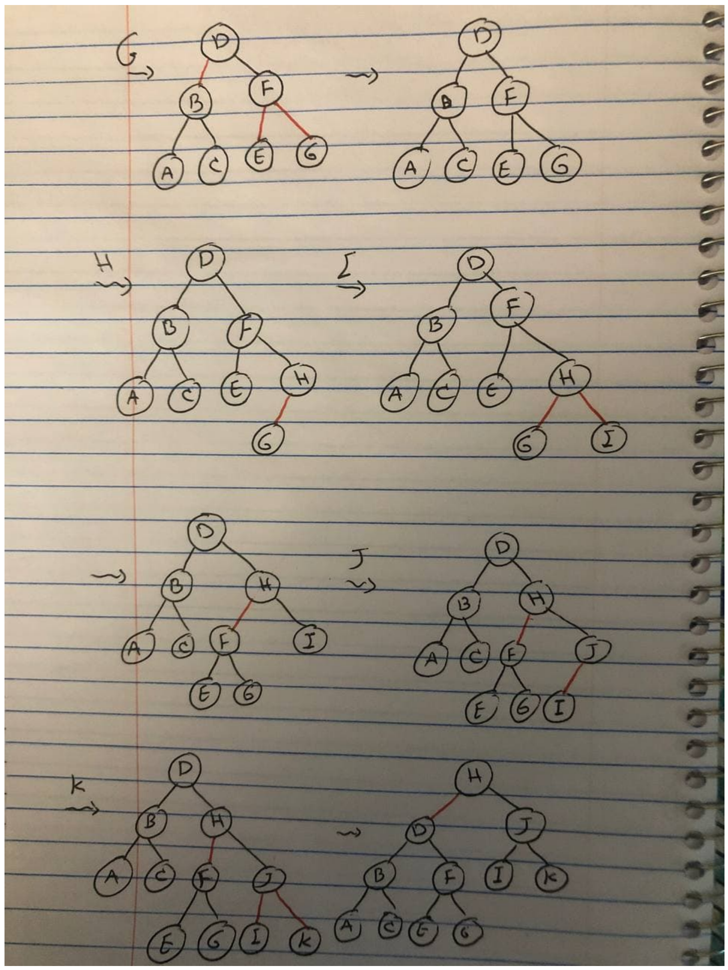

Figure 7: Question 7 (b)

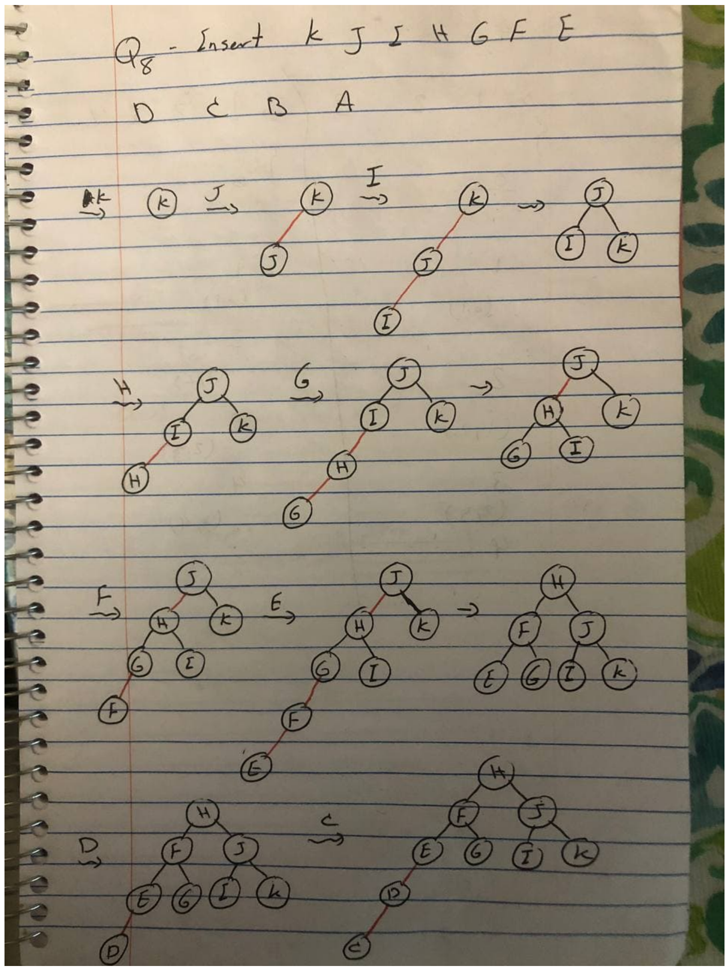

Figure 8: Question 8 (a)

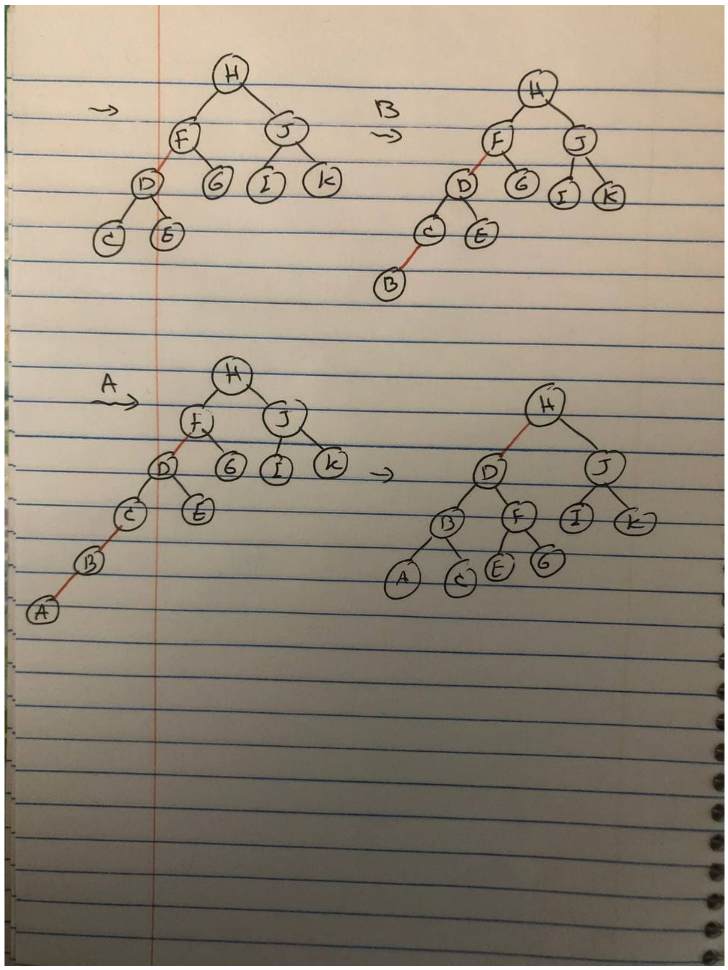

Figure 9: Question 8 (b)