CSc 225 Assignment 5: Trees

Due date:

The submission deadline is 11:55pm on Monday, July 20th, 2020

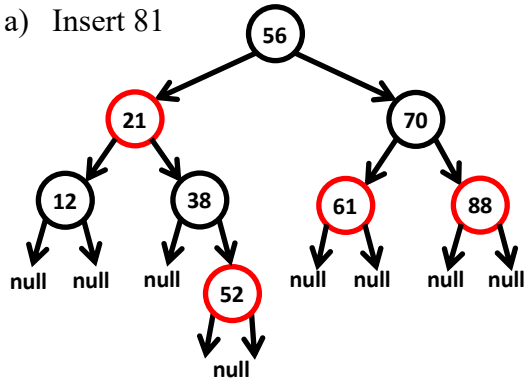
How to hand it in:

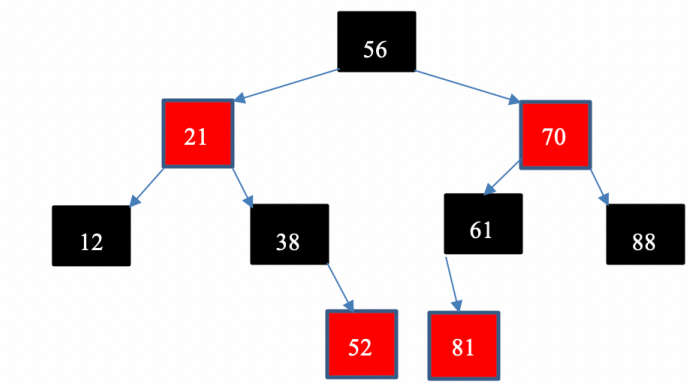
Submit a **.pdf** file (Part I and Part II) and the **`HuffmanTree.java`** file(for Part 3) through the Assignment 5 link on the CSC225 ConneX page.

Part 1: Red-Black Trees

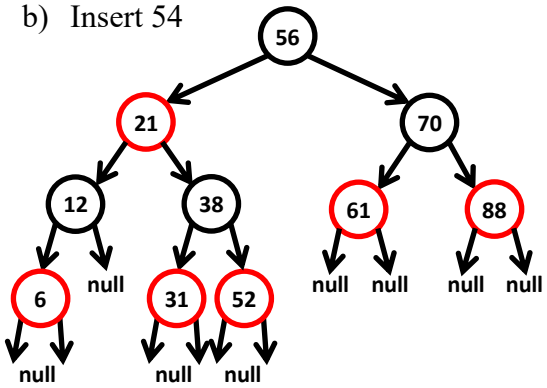
1. Draw the completed **Red-Black** Tree after the specified insertion.

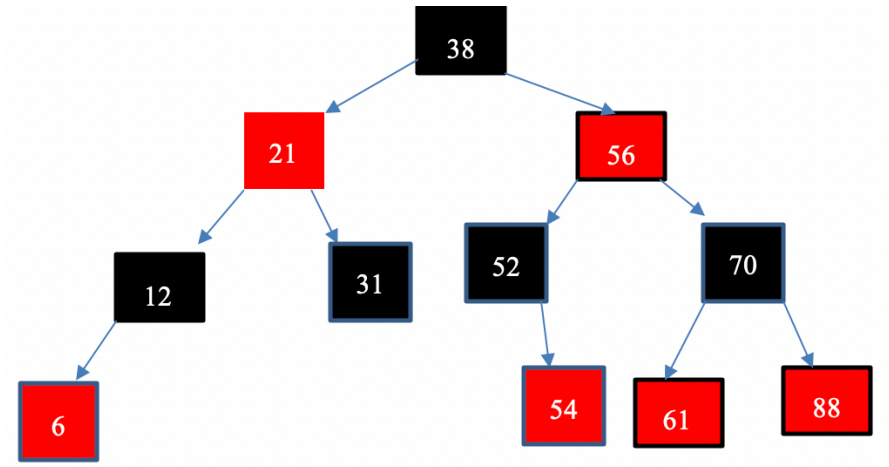
a) Insert 81





b) Insert 54

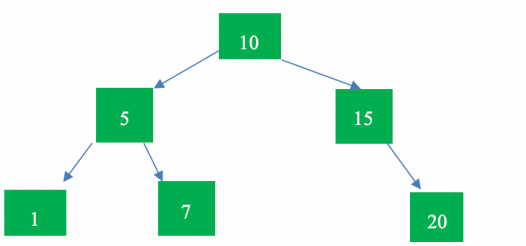




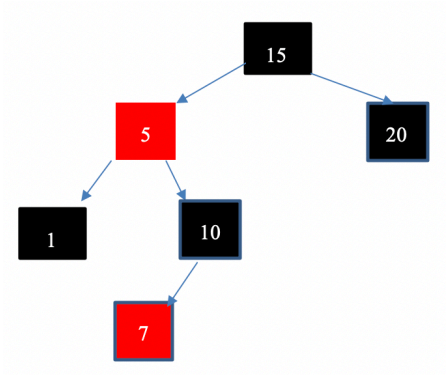
2. Determine a sequence of keys to insert into a BST and a Red-Black Tree such that the height of the BST is less than the height of the Red-Black Tree, or prove that no such sequence is possible.

   10,15,20,5,1,7. BST Height = 2, RBT Height = 3.

BST:



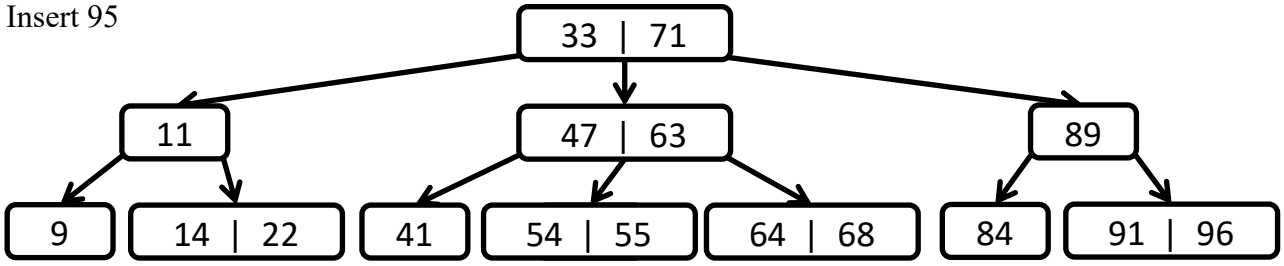
RBT:

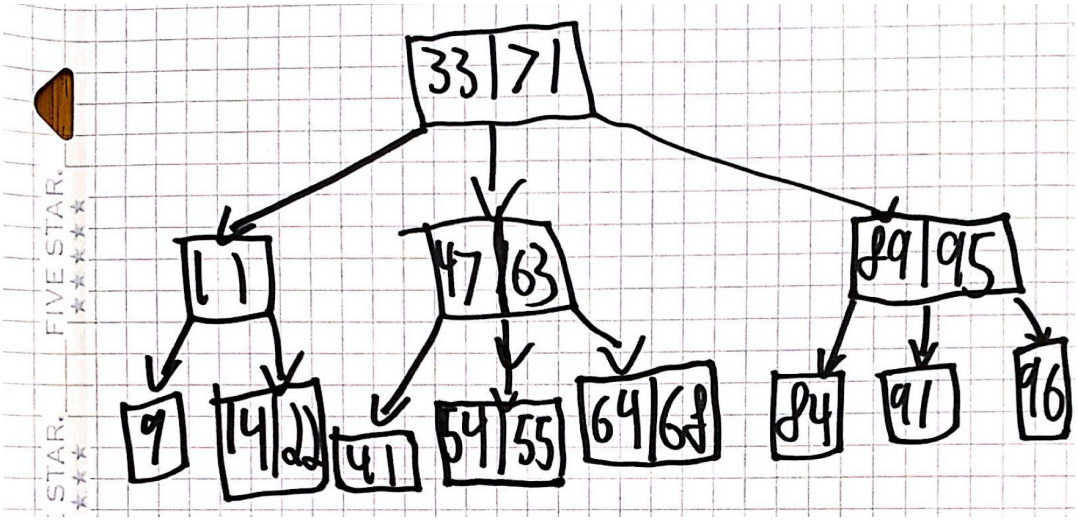


Part 2: B-Trees

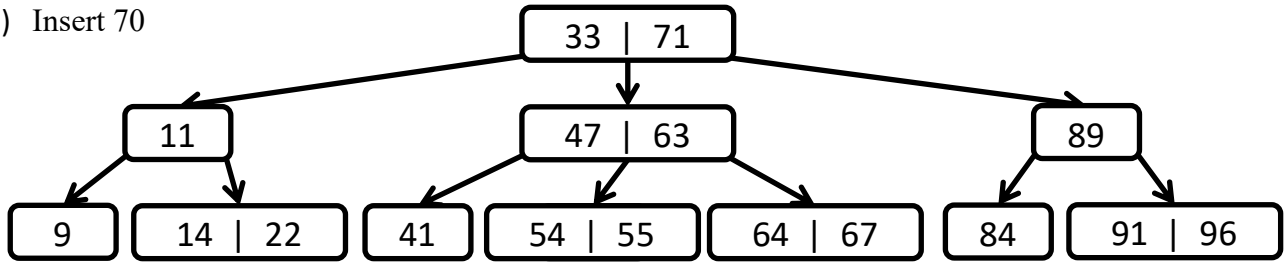
3. Draw the completed **2-3 Tree** after each of the specified insertion.

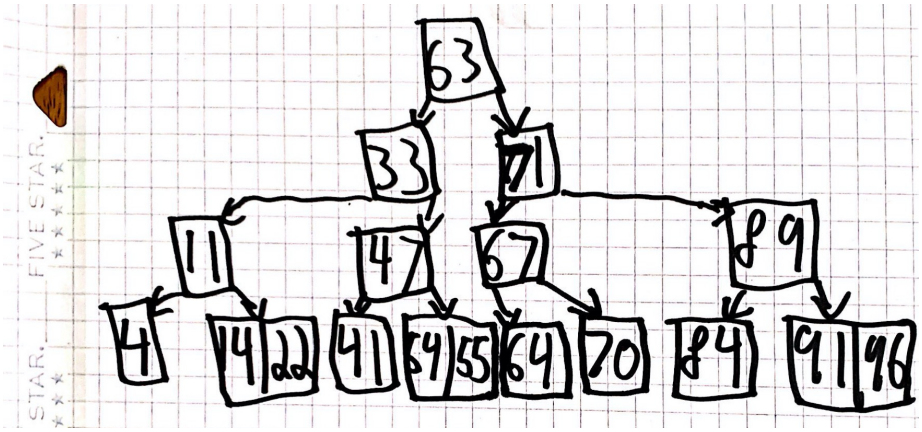
a) Insert 95





b) Insert 70





4. Given a B-Tree with $m = 4$, what is the minimum number of elements that could be inserted for which the tree would have a height of 2 (root, plus two levels below). Briefly explain how this scenario would occur (providing example insertions sequence would suffice).

   10 would be the minimum amount of elements to insert to get b tree (m=4) a height of 2.

   The sequence to achieve this would be : 1,2,3,4,5,6,7,8,9,10. (Or any digits in increasing order). This method only works if the left of center element is chosen as the element to move up during an over flow.

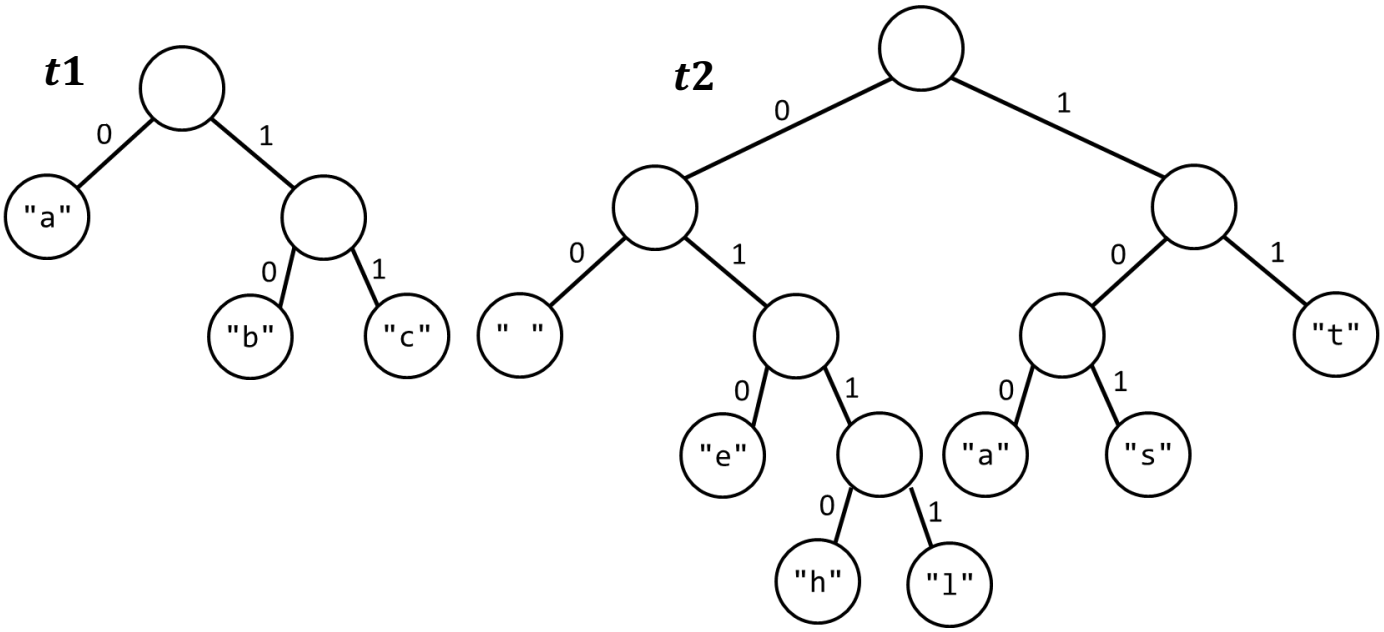
5. Given a B-Tree with $m = 4$, what is the maximum number of elements that could be inserted for which the tree would have a height of 2 (root, plus two levels below). Briefly explain how this scenario would occur (what would the contents of the nodes look like).

   63 elements is the maximum number of elements. This occurs when all nodes are completely full (3 elements) and all nodes have 4 children. 1(root) + 4(children of root) + 16(grandchildren) = 21 nodes times 3 elements each = 63 elements. The sequence of number also must fill all leaf nodes with 3 elements before adding a fourth element which creates the overflow.

Part 3: Implementation

6. For Part 3 you will be decoding bits of data using a Huffman tree, as shown in lecture. Two Huffman trees are provided for you; your task is to implement the decode method so that a textual representation can be obtained from a given input bit string.

   The two Huffman trees are shown below:



   Examples:
   - In $\boldsymbol{t1}$, the letter b is obtained by a 10 encoding.
   - In $\boldsymbol{t2}$, the letter s is obtained by a 101 encoding
   - In $\boldsymbol{t2}$, the encoding 1010110010 could be decoded to “she” (101=s, 0110=h, 010=e)

   Download the a5_files.zip file containing all of the starter files for this programing component of this assignment. The file can be found in the Resources > Assignments > a5 section on ConneX.

   The **`A5Tester.java`** file has tests for the two Huffman trees $t1$ and $t2$ with some example encodings. Once you have completed the decode method, submit the **`HuffmanTree.java`** file.