**Activity – Trees (50 pts)**

Given an arbitrary list of **strings**, create a **"reversed"** binary search tree (BST) (i.e., **the maximum value is on the leftmost subtree and the minimum value is at the rightmost subtree**) that is capable of the following operations:

* Creating the tree
* Searching through the tree and displaying the path from the root to the searched node
* Deleting a node from the tree

Make sure that error handling is implemented.

The script should contain the following features:

1. The user should input a sequence of (space-delimited) strings.

Inserting image...

Enter a sequence of strings (space-delimited): abk abq abt abr abw abz abu abv

1. Based on the user input, the script should create a “reversed” BST and display the generated BST akin to a directory structure. The number of dashes represents the level wherein the node occupies (*level + 1* from the lecture).

(ROOT ) |\_ abk

(LEFT ) |\_\_ abq

(LEFT ) |\_\_\_ abt

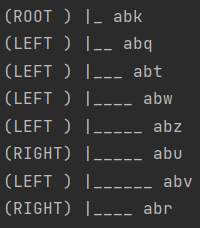
(LEFT ) |\_\_\_\_ abw

(LEFT ) |\_\_\_\_\_ abz

(RIGHT ) |\_\_\_\_\_ abu

(LEFT ) |\_\_\_\_\_\_ abv

(RIGHT ) |\_\_\_\_ abr



1. The user should input a string that will be searched in the tree.



Enter string to search for in the tree: abv

1. If the string is found, then the script should output the traversed path from the root to the node.



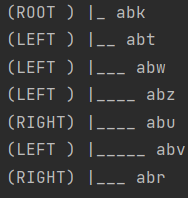
Path from abk to abv: abk -> abq -> abt -> abw -> abu -> abv

1. The user should input a string that will be deleted from the tree.



Enter string to delete in the tree: abq

1. If the string exists, then the script should delete the node, adjust the connections based on the BST deletion criteria (taking note that the BST is reversed), and display the resulting tree.



(ROOT ) |\_ abk

(LEFT ) |\_\_ abt

(LEFT ) |\_\_\_ abw

(LEFT ) |\_\_\_\_ abz

(RIGHT ) |\_\_\_\_ abu

(LEFT ) |\_\_\_\_\_ abv

(RIGHT ) |\_\_\_ abr