Name: Inho choT **ID:** 180188.

Problem 1

Suppose you have a Binary Search Tree with N nodes whose preorder traversal is the same as its inorder traversal. What is the cost for finding the Min value from such a tree? Max value?

Min : O(1)

Max : O(N).

Problem 2

You want to maintain peoples' names and for each person, keep track of all the streets they lived on. Assume there are P total people and each person has lived on average E former streets. Assume you're using a map of name/streets : map< names, set< streets >> names2streets;

What is the Big-O cost of:

a. Finding the names of all people who lived on "Levering Street"?

$O(P * \log_2 E)$.

b. Determining if "Bill" ever lived on "Westwood Blvd"?

$O(\log_2 P * \log_2 E)$.

c. Printing out every name along with each person's street addresses in alphabetical order?

$O(PE)$.

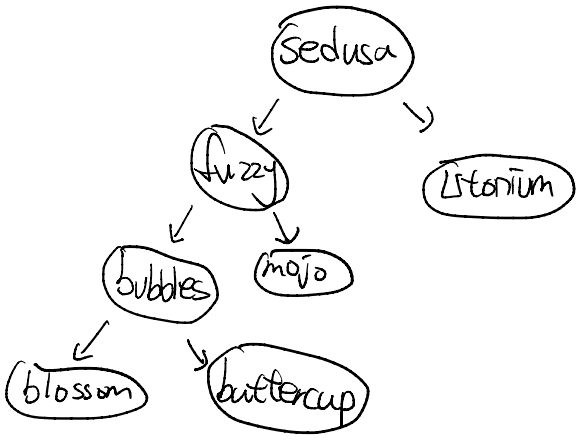
d. Printing out all the streets that "Tala" has lived on?

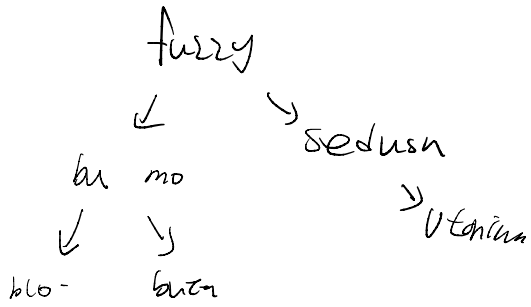
$O(\log_2 P + E)$.

Name: Inho Choi **ID:** 1801n8n.

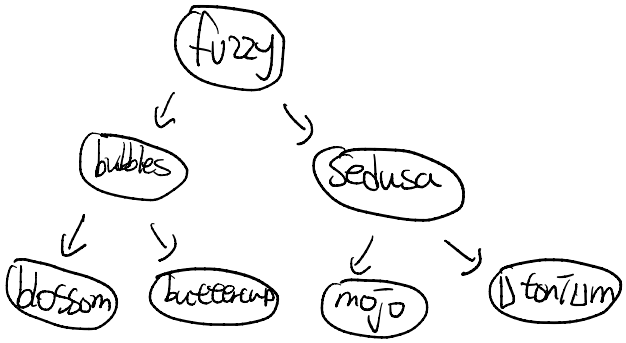
Problem 3:

a. Draw a binary search tree that is created if the following strings are inserted into the tree in the given order: “sedusa”, “fuzzy”, “bubbles”, “utonium”, “buttercup”, “mojo”, “blossom”.





b. Draw a balanced binary search tree containing the same names given above (Just draw what the tree looks like, do not attempt a balancing algorithm).



c. How would you describe the Big-O for searching a tree with a structure similar to the tree in part a? part b?

Part a's Big-O is $O(\log_2 n)$.

Part b's Big-O is $O(\log_2 n)$ because balanced tree can always be searched in $\log_2 n$ time

Name: Inho choi **ID:** 1801n8n.

Problem 6:

Consider the following Binary Search Tree, Show what the tree looks like after deleting the root node.