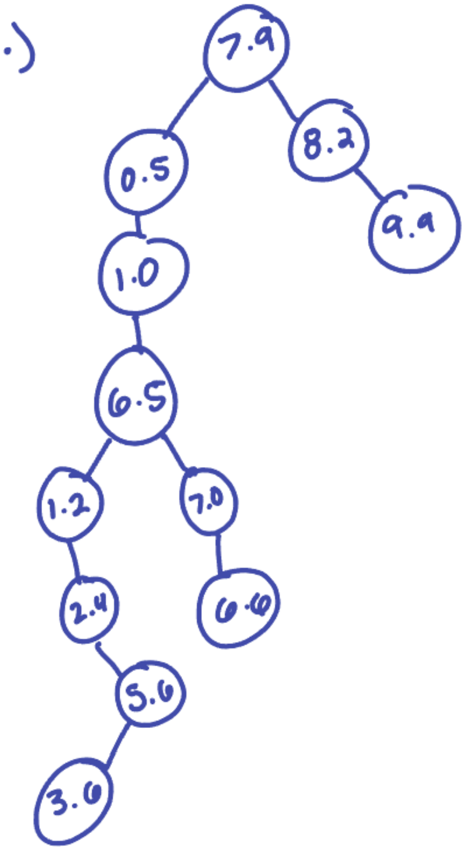
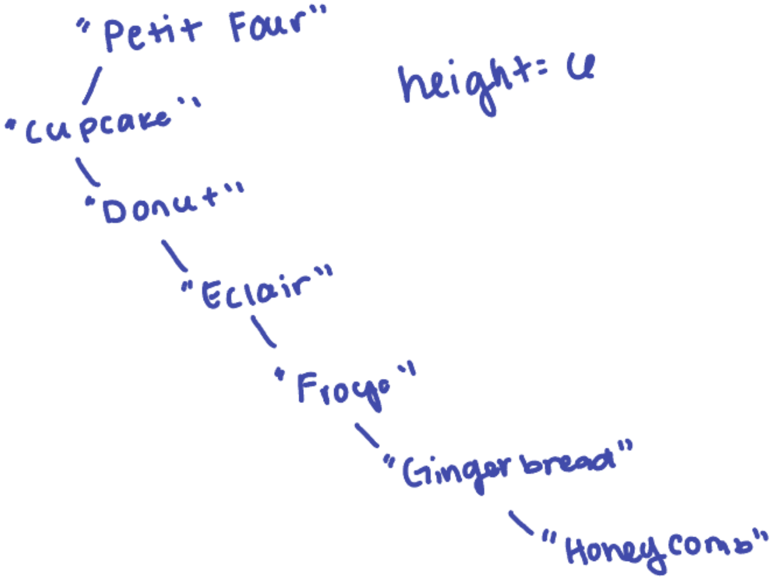
Question 1:

a.)



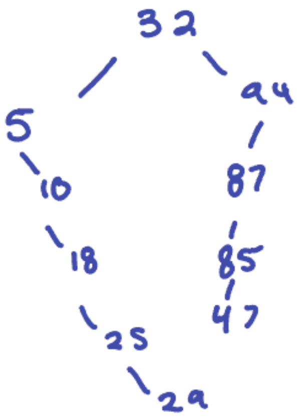
height=7

b.)



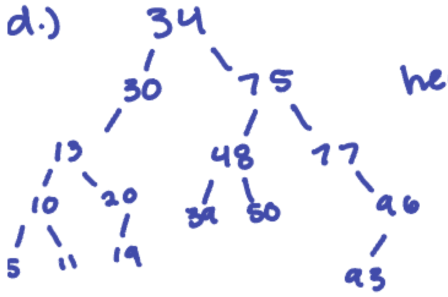
height: 6

c.)



height=5

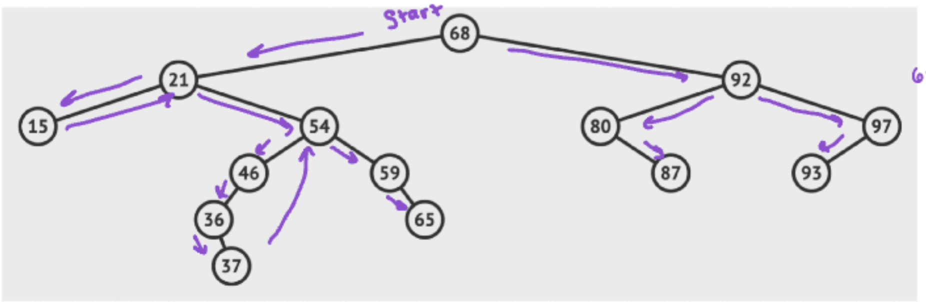
d.)



height=4

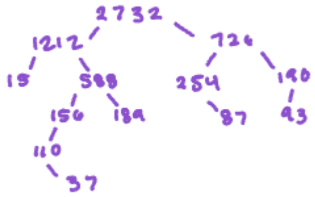
Question 2:

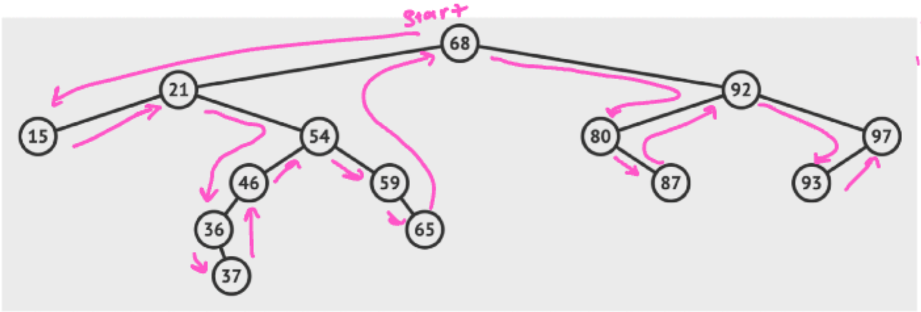
a.)



Preorder

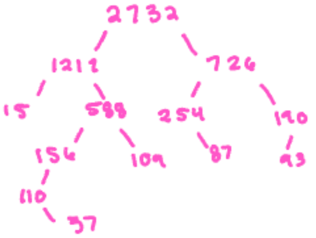
68, 21, 15, 54, 46, 36, 37, 59, 65, 92, 80, 87, 97, 93

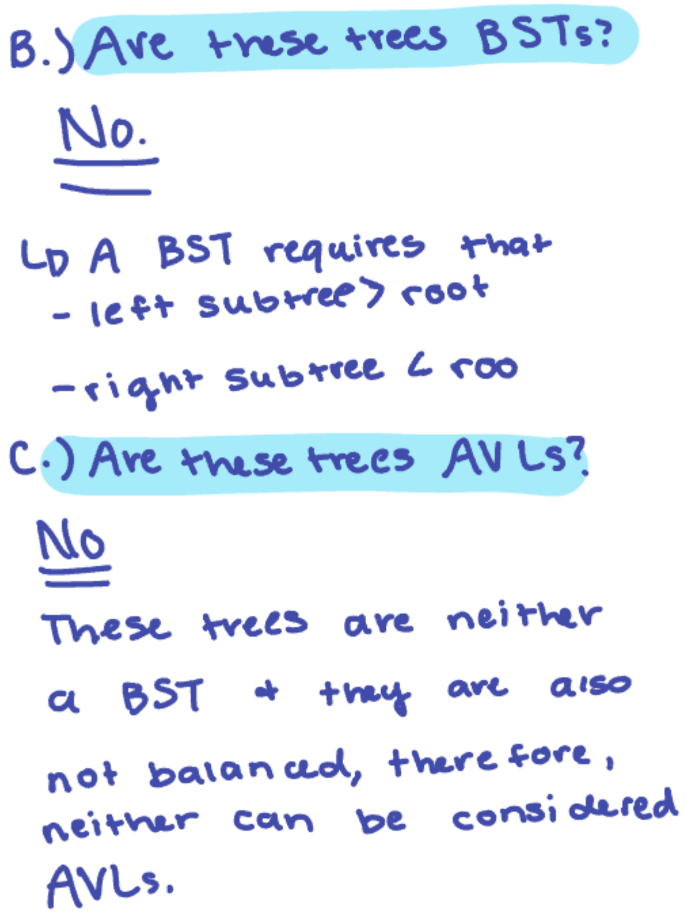




Inorder

15, 21, 36, 37, 46, 54, 59, 65, 68, 80, 87, 92, 93, 97





B.) Are these trees BSTs?

No.

↳ A BST requires that

- left subtree > root
- right subtree < roo

C.) Are these trees AVLs?

No

These trees are neither a BST + they are also not balanced, therefore, neither can be considered AVLs.

Question 3:

Link to my Github repo: https://github.com/sloaneeliza/TreesAndHeaps-Assignment6/tree/main

Question 4:

- Zybooks question

Question 5:

- Election.java
    - Time complexity: Overall complexity is O(n log n) where n is the number of candidates. This is due to the sorting operation in getTopKCandidates (which uses a PriorityQueue) and auditElection.
    - Space complexity: O(n) where n is the number of candidates. The space requirement comes from storing the list of candidates in an ArrayList and their vote counts in a HashMap.
- ElectionSystem.java
    - Time complexity: O(n log n) inherited from the Election class, we get O(n log n) from the Election class methods getTopKCandidates and auditElection. n is the number of candidates
    - Space complexity: O(n) where n is the number of candidates. The space complexity for ElectionSystem depends on the Election object we use, which has a time complexity of O(n).