4)

a) Balanced Binary Search Tree properties:

- Each node can have at most two nodes
- For every node is the largest node value of its left subtree
- Every node is the smallest node value of its right subtree
- Every node must maintain a balance factor of less than abs(1)

B)

Function	Worst Coase	Notes *n → number of nodes in tree
Insert	O(logn)	B/c tree is balanced finding insertion point → O(logn) Balancing → O(logn) O(logn) + O(logn)
Delete	O(logn)	finding node to remove → O(logn) balancing tree O(log n)
Find-next	O(logn)	balanced So Search will be O(logn)
Find-prev	O(logn)	balanced So Search will be O(logn)
Find-min	O(logn)	balanced So Search will be O(logn)

Find-max	O(logn)	balanced So Search will be O(logn)

5a) Code

5b) When extremely large sets to inset into the tree – the recursive implementation had a stack overflow error.

This is because of the implicit space associated with method calls. With an iterative solution everything is contained in one call. On the other hand, with the recursive implementation with large sets we get so many recursive calls that must all be placed on a stack that we eventually run out of space and the program fails.

6b)

Starting Insert

BST iterative: 144367

BBST Iterative: 111324

The balanced tree took many less traversals down the tree than the iterative implementation did

6c)

Starting Insert

BST iterative sorted: 49985001

BBST Iterative sorted: 113618