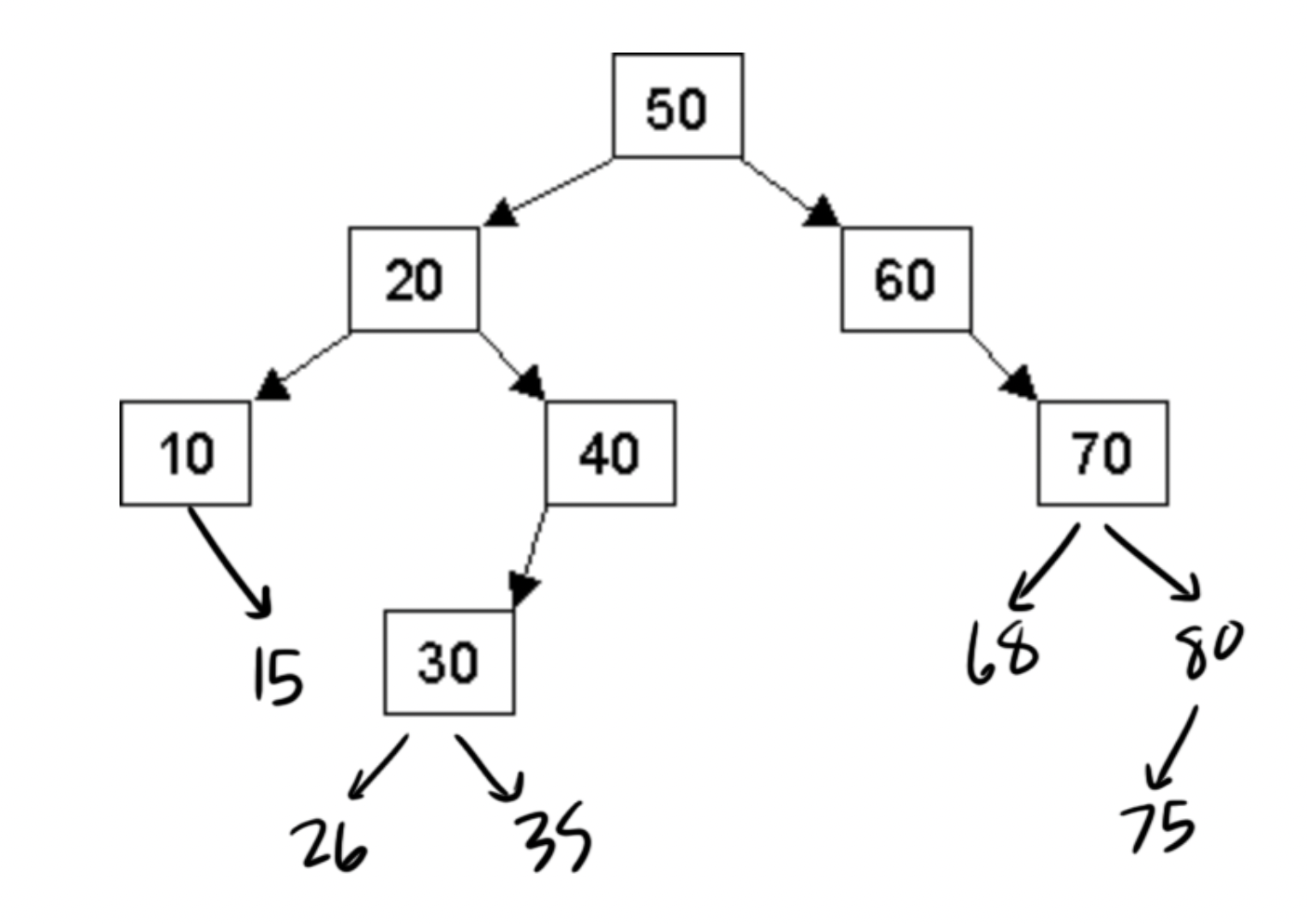
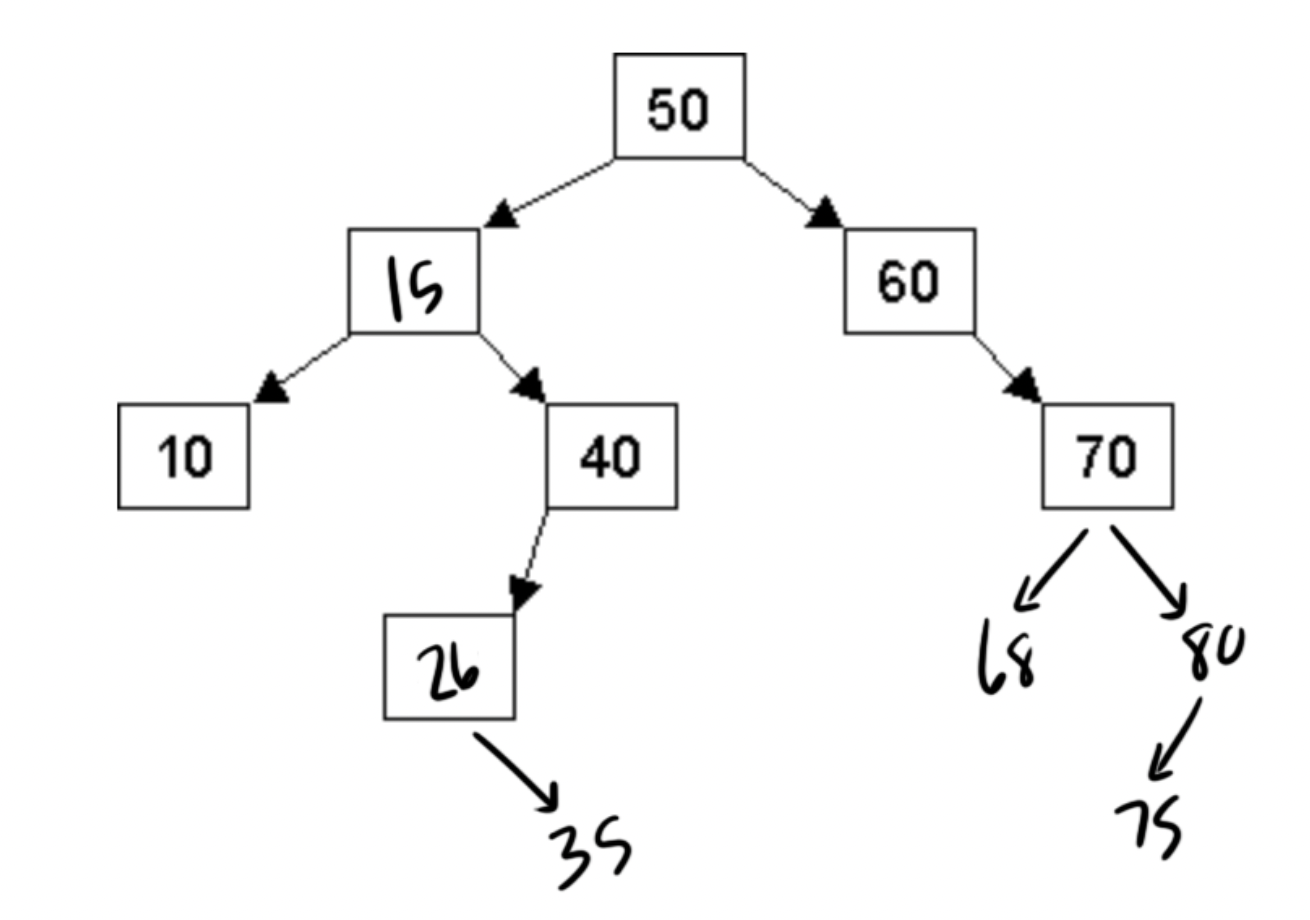
hw

1a.

1b. Inorder: 10 15 20 26 30 35 40 50 60 68 70 75 80

Preorder: 50 20 10 15 40 30 26 35 60 70 68 80 75

Postorder: 15 10 26 35 30 40 20 68 75 80 70 60 50

1c.

2a.

struct Node

{

int data;

Node \*left, \*right, \*parent;

};

2b.

// Inserts a new Node into a BST with parent pointers

// Inserting a value already in the tree does nothing

// Returns true if the insertion is successful, or false if not

bool add(int newEntry)

create a new node with the given value

if the tree is empty

set root pointer to the new node

set left, right, and parent pointers to nullptr

return true

start at the root of the tree

while the current node isn’t a nullptr

if data in our new node is equal to the value in the current node

delete the created node

return false

if data in the new node is less than the value in the current node

if there is a left child

go left to the left child

else

set the current node's left child to the new node

set the new node's parent to the current node

return true

if data in the new node is greater than the value in the current node

if there is a right child

go right to the right child

else

set the current node's right child to the new node

set the new node's parent to the current node

return true

return false

3a.

3b. 7 5 6 1 0 3

3c. 6 5 3 1 0

4a. O(C + S)

4b. O(logC + S)

4c. O(logC + logS)

4d. O(logS)

4e. O(1)

4f. O(logC + S)

4g. O(SlogS)

4h. O(ClogS)