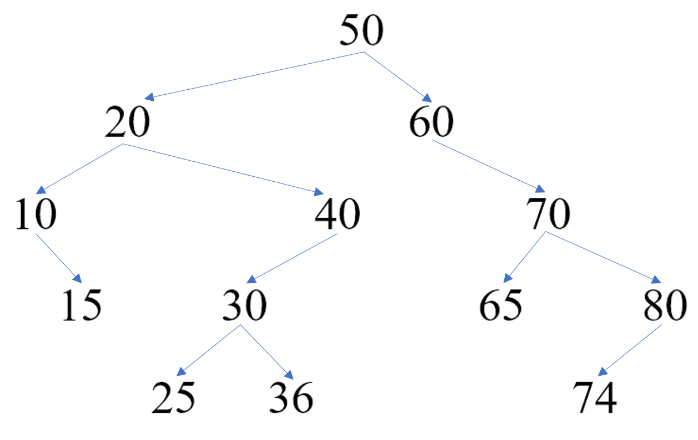
1.

a. after insertion:



b.

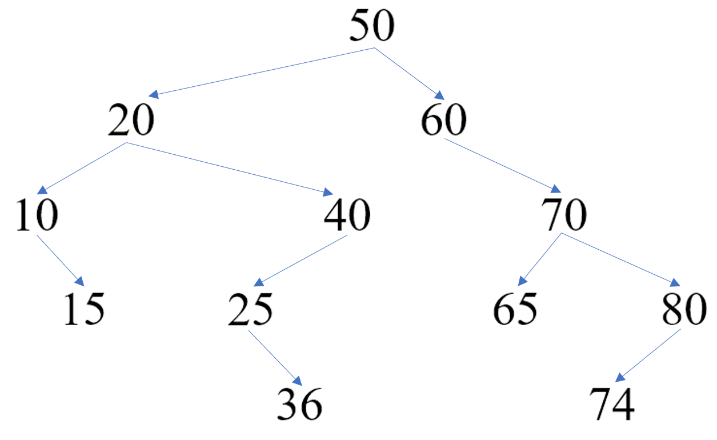
pre-order: 50 20 10 15 40 30 25 36 60 70 65 80 74;

in-order: 10 15 20 25 30 36 40 50 60 65 70 74 80;

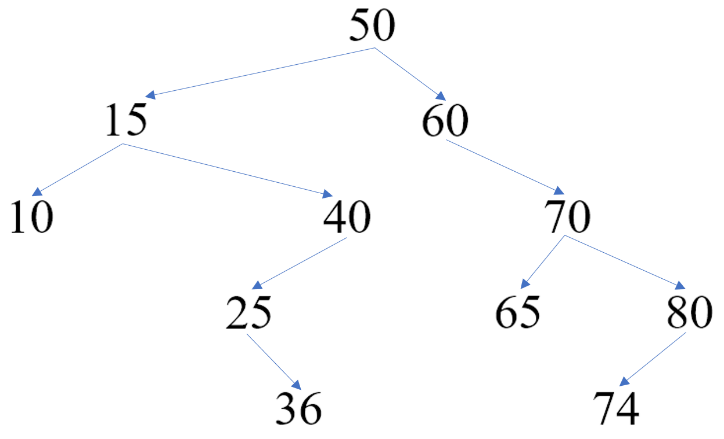
post-order: 15 10 25 36 30 40 20 65 74 80 70 60 50.

c.

i) after deleting 30:



ii) after deleting 20:



2.

a.

struct Node

{

int value;

Node\* leftChild;

Node\* rightChild;

Node\* parent;

};

b.

if root == nullptr

create a new node pointed by temp with input value (both children are null pointers)

root = temp

root->parent = nullptr

return

create a node pointer p (assigned it root) and an auxiliary pointer q

while p is not nullptr, repeatedly:

q = p

if input value < p->value

p = p->leftChild

else if input value > p->value

p = p->rightChild

else // input value == p->value

return // do nothing else

create a new node pointed by temp with input value (both children are null pointers)

temp->parent = q // update parent

if input value < q->value // update child

q->leftChild = temp;

else // input value > q->value

q->rightChild = temp

return

3.

a.

8

3 6

0 2 4

b.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 8 | 3 | 6 | 0 | 2 | 4 |

c.

6

3 4

0 2

4.

a. O (C + log S)

b. O (log C + S)

c. O (log C + log S)

d. O (log S)

e. O (1)

f. O (log C + S)

g. O (S log S)

h. O (C log S)