DSAA Ass 9 12310401 王子恒.

Q9.1

1. Base: when $h=0$ the complete binary tree has only one root node and no internal node, thus $0 = 2^0 - 1 = 1 - 1 = 0$

Inductive Hypothesis: If a k height complete binary tree has $2^k - 1$ internal nodes, then a $k+1$ height complete binary tree has $2^{k+1} - 1$ internal nodes.

Induction step: By the properties of complete binary trees, a tree of height $k+1$ can be seen as composed of two trees of height k. By the inductive hypothesis, each c.b tree of height has $2^k - 1$ internal nodes. Therefore, the two trees together have $2 \times (2^k - 1)$ internal nodes. Adding the new root node, we have total $2 \times (2^k - 1) + 1 = 2^{k+1} - 1$ internal nodes. Thus proves that a complete binary tree of height $k+1$ has $2^{k+1} - 1$ internal nodes.

2. Base: when the tree has only one node, it is also the leaf node. And it has no internal nodes. It satisfy the leaf nodes is one more than the internal nodes.

Inductive Hypothesis: If a h height full nonempty binary tree, the number of leaf nodes is one more than the internal nodes, then it holds the property for $h+1$ height full nonempty binary tree.

Inductive Step: Consider a full nonempty binary tree. By the inductive hypothesis, the number of leaves is one more than the internal nodes. When we add a new level to the tree, each leaf node will generate two new leaf nodes, and one new internal nodes. Therefore, the increase of leaves is equal to the increase of internal nodes. Thus, the new full binary tree still have leaves one more than the intervals.

3. Base: when $h=0$ $|V|=1$ $|E|=0$ so $|V|=|E|+1$

Inductive hypothesis: If for a nonempty binary tree, $|V|=|E|+1$, then it still holds if add a new nodes.

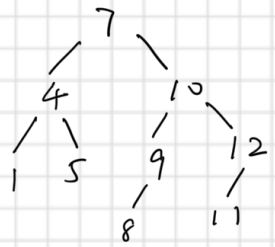
Induction Step: When we add a new node, the $|V|$ will increase 1 and $|V|$ also increase 1. thus $|V|=|E|+1$ still holds.

Q9.2 1. 11 > 7 7.right = 10

11 > 10 10.right = 12

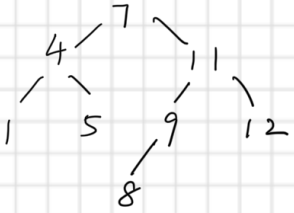
11 < 12 12.left = NIL

12.left = 11.



2. successor is 11 of the right tree

transplant 10 with 11

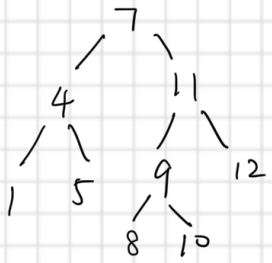


3. 10 > 7 7.right = 11

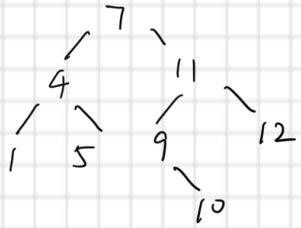
10 < 11 11.left = 9

10 > 9 9.right = NIL

9.right = 10



4. 8 is leaf node



5. succesor is 9 of the right tree.

Transplant 9 with 10

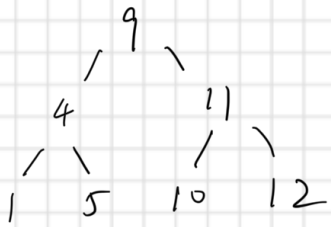
y.right = z.right = 11.

11.parent = 9.

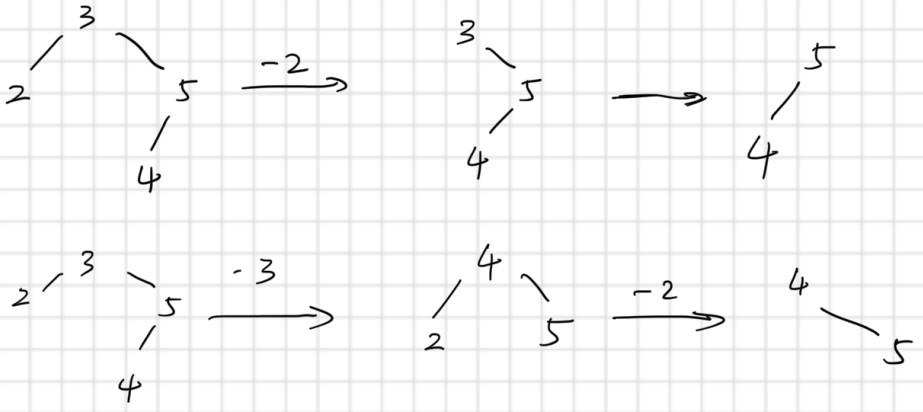
Transplant 7 with 9

9.left = 7.left = 4

4.p = 9



Q9.3 Yes the result tree can be different



Q9.4. Tree - Predecessor (x)

```
if x. left is not NIL
    return Tree-Maximum (x.left)

else
    y = x.p
    while y is not NIL and x == y.left
        x = y
        y = y.p
    return y
```

Q9.5

For the worst case, when the set is in order, every time insert an element should scan all the elements before, cost $O(n)$ thus the build time of the BST is $n \cdot O(n) = O(n^2)$

For the best case, the sort is balanced, then it cast $O(\log n)$ every insert. The total build time is $O(n \log n)$

the Inorder walks output all elements in $\Theta(n)$

Thus the worst case $O(n^2) + O(n) = O(n^2)$

the best case $O(n \log n) + O(n) = O(n \log n)$