1. Before:

Case1

a）15

b) **if** node.\_left **is not** None:

**return** self.\_subtree\_last\_position(node.\_left)

c) we first check 15 and find it less than 18, so we go on the right branch and compare it with 23, we found it is bigger than 18, so we get to the left branch and check it with 20. Then we check it with 20 and find it still larger than 18 so we check the left branch of it and find 18. Because we find that 18 has no left side and when we check its parent side, we find all are at the left side except 15. As a result, we know that 15 is before (18).

Case2:

a) 66

b) **else**:

walk = node

above = walk.\_parent

**while** above **is not** None **and** walk == above.\_left:

walk = above

above = walk.\_parent

c) Because it is before, so we first check the left side and find 16 then we check if 16 has right side and we find 21. Repeat the step and we find 65 and finally at 66 we find that there is no more right-side element existed. As a result, we return 66 as before (69)

After:

Case1:

a)35

b) **if** node.\_right **is not** None:

**return** self.\_subtree\_first\_position(node.\_right)

c)We first find out where 23 is, and check if it has the right element and find 64. Then we are looping about the left element of the element and find that 35 is the element that has no more left element, therefore 35 is the answer

Case2:

a)67

b) **else**:

walk = node

above = walk.\_parent

**while** above **is not** None **and** walk == above.\_right:

walk = above

above = walk.\_parent

c)We first find the place where 66 is and find that there is no right element of 66, so we trace back to the element where it shows 66 is at the left side, Therefore we find 67

delete:

1. 20

/ \

12 45

/ \

2 49

\

89

/

63

b) **else**:

parent = node.\_parent

**if** node **is** parent.\_left:

parent.\_left = child

**else**:

parent.\_right = child

c) we first find out 17 then delete it. Then we find that because 17 has no left or right element, so we just delete it.

Case2:

1. 84

/

41

/ \

34 50

/ / \

10 46 55

b) **else**:

parent = node.\_parent

**if** node **is** parent.\_left:

parent.\_left = child

**else**:

parent.\_right = child

c)We find that 1 is the left element of 84 and a only have one right element. As a result we only need to let 41 become the new left element of 84

Case 3

1. 73

/ \

23 92

/ \ / \

17 64 87 99

/ \ / \

53 66 75 89

/\ / /\

46 63 65 7477

/ \

35 80

b) **if** node **is** self.\_root:

self.\_root = child

c) We check which part is the cloest child of the root and make the child as the new root.