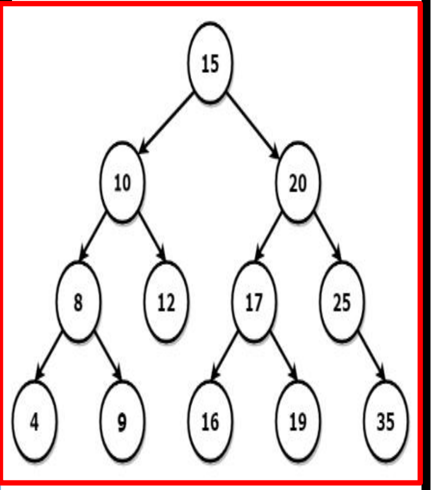
Case1: Delete a child node

STACK 1	STACK 2
main()	delete(ptr, 12) ->ptr is 15
delete(root,12)	if(ptr==NULL) -> False
	else if(12<15) -> True
	ptr->left = delete(ptr->left,12)

STACK 3	STACK 4
delete(ptr, 12) -> ptr is 10	delete(ptr, 12) -> ptr is 12
if(ptr==NULL) ->False	if(ptr==NULL) ->False
else if(12<10) -> False	else if(12<12) -> False
else if(12>10) -> True	else if(12>12) -> False
ptr->right = delete(ptr->right,12)	else -> True
	if(ptr->left &&ptr->right) ->False
	else -> temp = ptr
	if(ptr->left == NULL) ->True
	ptr = ptr->right
	free(temp) -> delete the node 12
	return ptr to prev recursive call

INPUT TREE

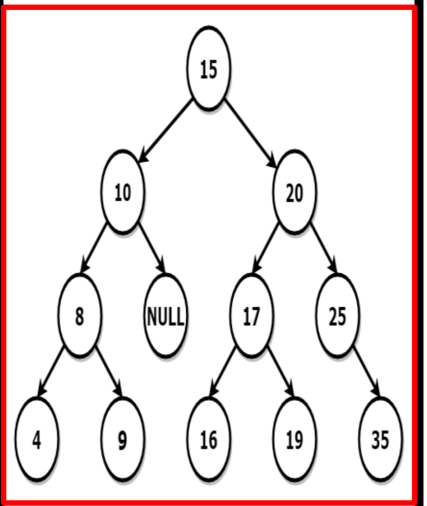


Case2: Delete a child with one child

STACK 1	STACK 2
main()	delete(ptr, 25) ->ptr is 15
delete(root,25)	if(ptr==NULL) -> False
	else if(25<15) -> False
	else if(25>15) -> True
	ptr->right = delete(ptr->right,25)
	15->right becomes 20
	return node 15 to main function

STACK 3	STACK 4
delete(ptr, 25) -> ptr is 20	delete(ptr, 25) -> ptr is 25
if(ptr==NULL) ->False	if(ptr==NULL) ->False
else if(25<20) -> False	else if(25<25) -> False
else if(25>20) -> True	else if(25>25) -> False
ptr->right = delete(ptr->right,25)	else -> True
	if(ptr->left &&ptr->right) ->False
20->right becomes 35	else -> temp = ptr
return the node 20 to prev recursive call	if(ptr->left == NULL) ->True
	ptr = ptr->right = points to node 35
	free(temp) -> delete the node 25

INPUT TREE



return ptr to prev recursive call