def rebalance (self, node\_to\_rebalance):

self.rebalance\_count += 1

A = node\_to\_rebalance

F = A.parent #allowed to be NULL

if node\_to\_rebalance.balance() == -2:

if node\_to\_rebalance.rightChild.balance() <= 0:

"""Rebalance, case RRC """

B = A.rightChild

C = B.rightChild

assert (not A is None and not B is None and not C is None)

A.rightChild = B.leftChild

if A.rightChild:

A.rightChild.parent = A

B.leftChild = A

A.parent = B

if F is None:

self.rootNode = B

self.rootNode.parent = None

else:

if F.rightChild == A:

F.rightChild = B

else:

F.leftChild = B

B.parent = F

self.recompute\_heights (A)

self.recompute\_heights (B.parent)

else:

"""Rebalance, case RLC """

B = A.rightChild

C = B.leftChild

assert (not A is None and not B is None and not C is None)

B.leftChild = C.rightChild

if B.leftChild:

B.leftChild.parent = B

A.rightChild = C.leftChild

if A.rightChild:

A.rightChild.parent = A

C.rightChild = B

B.parent = C

C.leftChild = A

A.parent = C

if F is None:

self.rootNode = C

self.rootNode.parent = None

else:

if F.rightChild == A:

F.rightChild = C

else:

F.leftChild = C

C.parent = F

self.recompute\_heights (A)

self.recompute\_heights (B)

else:

assert(node\_to\_rebalance.balance() == +2)

if node\_to\_rebalance.leftChild.balance() >= 0:

B = A.leftChild

C = B.leftChild

"""Rebalance, case LLC """

assert (not A is None and not B is None and not C is None)

A.leftChild = B.rightChild

if (A.leftChild):

A.leftChild.parent = A

B.rightChild = A

A.parent = B

if F is None:

self.rootNode = B

self.rootNode.parent = None

else:

if F.rightChild == A:

F.rightChild = B

else:

F.leftChild = B

B.parent = F

self.recompute\_heights (A)

self.recompute\_heights (B.parent)

else:

B = A.leftChild

C = B.rightChild

"""Rebalance, case LRC """

assert (not A is None and not B is None and not C is None)

A.leftChild = C.rightChild

if A.leftChild:

A.leftChild.parent = A

B.rightChild = C.leftChild

if B.rightChild:

B.rightChild.parent = B

C.leftChild = B

B.parent = C

C.rightChild = A

A.parent = C

if F is None:

self.rootNode = C

self.rootNode.parent = None

else:

if (F.rightChild == A):

F.rightChild = C

else:

F.leftChild = C

C.parent = F

self.recompute\_heights (A)

self.recompute\_heights (B)