# Test Data

### Path Constraints

getRoot():

Use on RBTreeWithFaults: RBTreeWithFaults

RBTreeWithFaults ():

None

RBTreeWithFaults (RBNode root):

None: root = new RBNode(params)

createInfinityNode ():

None:

createInfinityNode (RBNode root):

None root = new RBNode(params)

createNullNode(RBNode parent):

None parent = this.Root

search(**int** k)

DEPENDENT ON SEARCHNODE

insert(int k, string v)

Dependent on search, new RB node, fix up tree

leftChild(RBNode x,RBNode y)

x!=null, y!=null x= new RBNode(params) y = new RBNode(params)

transplate(RBNode x, RBNode y)

x!=null, y!=null, x==x.Parent.Left x= new RBNode(params).parent.left, y = new RBNode(params)

x!=null, y!=null, x!=x.Parent.Left x= new RBNode(params).parent.right, y = new RBNode(params)

rightRotate(RBNode y)

y!=null, y.Left!=null

deleteFixup(RBNode x,RBNode y)

dependent on many things

minimumNode(RBNode node)

node!=null, node.Key==integer.MAX\_VALUE

node!=null, node.Key==integer.MAX\_VALUE, , node.Left.Key==integer.MAX\_VALUE

node!=null, node.Key!=integer.MAX\_VALUE, node!=null, node.Left.Key!=integer.MAX\_VALUE

isNullNode(RBNode node)

node!=null

maxValue(RBNode node)

node!=null, node.Right.Key==integer.MAX\_VALUE

node!=null, node.Right.Key!=integer.MAX\_VALUE

ArrayOfStringsToArrayOfInts(String[] strArr)

strArr={“1”}

valuesToArray()

**this**.Root.Left!= null

Size()

**this**.Root.Left!=null

RBNode(String value,**int** key, RBNode left, RBNode right,RBNode parent)

None

## Data Definitions:

A,B,C,D,E= RBNode(“String”, {1-5}, given)

V,W,X,Y,Z= RBNode(“String”, {6-10}, given)

Tree= RBTreeWithFaults(given)