**Binary Tree**

Python:

|  |
| --- |
| # Python3 program to program to check if a  # given Binary Tree is symmetric or not    # Helper function that allocates a new  # node with the given data and None  # left and right pairs.  class newNode:        # Constructor to create a new node      def \_\_init\_\_(self, key):          self.key = key          self.left = None          self.right = None    # function to check if a given  # Binary Tree is symmetric or not  def isSymmetric( root) :        # if tree is empty      if (root == None) :          return True        # If it is a single tree node,      # then it is a symmetric tree.      if(not root.left and not root.right):          return True        q = []        # Add root to queue two times so that      # it can be checked if either one      # child alone is NULL or not.      q.append(root)      q.append(root)        # To store two nodes for checking      # their symmetry.      leftNode = 0      rightNode = 0        while(not len(q)):            # Remove first two nodes to          # check their symmetry.          leftNode = q[0]          q.pop(0)            rightNode = q[0]          q.pop(0)            # if both left and right nodes          # exist, but have different          # values-. inequality, return False          if(leftNode.key != rightNode.key):              return False            # append left child of left subtree          # node and right child of right          # subtree node in queue.          if(leftNode.left and rightNode.right) :              q.append(leftNode.left)              q.append(rightNode.right)            # If only one child is present          # alone and other is NULL, then          # tree is not symmetric.          elif (leftNode.left or rightNode.right) :              return False            # append right child of left subtree          # node and left child of right subtree          # node in queue.          if(leftNode.right and rightNode.left):              q.append(leftNode.right)              q.append(rightNode.left)            # If only one child is present          # alone and other is NULL, then          # tree is not symmetric.          elif(leftNode.right or rightNode.left):              return False        return True    # Driver Code  if \_\_name\_\_ == '\_\_main\_\_':        # Let us construct the Tree      # shown in the above figure      root = newNode(1)      root.left = newNode(2)      root.right = newNode(2)      root.left.left = newNode(3)      root.left.right = newNode(4)      root.right.left = newNode(4)      root.right.right = newNode(3)      if (isSymmetric(root)) :          print("The given tree is Symmetric")      else:          print("The given tree is not Symmetric")    # This code is contributed by  # Shubham Singh(Nafis Abir) |

**Output**

The given tree is Symmetric