1.

def postorder(ptr):

if ptr != None:

postorder(ptr.left)

postorder(ptr.right)

print('[%2d]' %ptr.data, end=' ')

def preorder(ptr):

if ptr != None:

print('[%2d]' %ptr.data, end=' ')

preorder(ptr.left)

preorder(ptr.right)

class Node():

def \_\_init\_\_(self, data=None):

self.data = data

self.left = None

self.right = None

def insert(self, data):

if self.data:

if data < self.data:

if self.left:

self.left.insert(data)

else:

self.left = Node(data)

else:

if self.right:

self.right.insert(data)

else:

self.right = Node(data)

else:

self.data = data

def inorder(self):

if self.left:

self.left.inorder()

print(self.data)

if self.right:

self.right.inorder()

tree = Node()

datas = [10, 5, 21, 9, 13, 28, 3, 4, 1, 17, 32]

for d in datas:

tree.insert(d)

print("前序走訪為:")

preorder(tree)

print(' ')

print("後序走訪為:")

postorder(tree)

2.

class Delete\_Node():

def deletenode(self, root, key):

if root is None:

return None

if key < root.data:

root.left = self.deletenode(root.left, key)

return root

if key > root.data:

root.right = self.deletenode(root.right, key)

return root

if root.left is None:

new\_root = root.right

return new\_root

if root.right is None:

new\_root = root.left

return new\_root

succ = self.max\_node(root.left)

tmp = Node(succ.data)

tmp.left = self.left\_node(root.left)

tmp.right = root.right

return tmp

def left\_node(self, node):

if node.right is None:

new\_root = node.left

return new\_root

node.right = self.left\_node(node.right)

return node

def max\_node(self, node):

while node.right:

node = node.right

return node

class Node():

def \_\_init\_\_(self, data=None):

self.data = data

self.left = None

self.right = None

def insert(self, data):

if self.data:

if data < self.data:

if self.left:

self.left.insert(data)

else:

self.left = Node(data)

else:

if self.right:

self.right.insert(data)

else:

self.right = Node(data)

else:

self.data = data

def inorder(ptr):

if ptr != None:

inorder(ptr.left)

print('[%2d]' %ptr.data, end=' ')

inorder(ptr.right)

tree = Node()

datas = [10, 5, 21, 9, 13, 28, 3, 4, 1, 17, 32]

for d in datas:

tree.insert(d)

inorder(tree)

delete = Delete\_Node()

delete.deletenode(tree, 17)

print(' ')

print("刪除節點17後:")

inorder(tree)