Code ตัวอย่าง **Lab 9 : Binary Search Tree & recursion ทบทวน**

class node:

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

self.data = data

self.left = left

self.right = right

#---------------------No class BST

def inOrder(r):

if r:

inOrder(r.left)

print(r.data, end = ' ')

inOrder(r.right)

def addi(r, data):

if not r:

return node(data)

else:

if data < r.data:

r.left = addi(r.left, data)

else:

r.right = addi(r.right, data)

return r

def add(r, data): # recursive add

if not r:

return node(data)

else:

if data < r.data:

r.left = add(r.left, data)

else:

r.right = add(r.right, data)

return r

def printSideWay(r, level):

if r:

printSideWay(r.right, level+1)

print(' ' \* 3 \* level, r.data)

printSideWay(r.left, level+1)

def height(r):

""" return height of node pointed by r"""

if not r:

return -1

else:

hl = height(r.left)

hr = height(r.right)

if hl>hr:

return hl+1

else:

return hr+1

def path(r, d):

"""print path from node pointed by r to node that has data d"""

if r.data != d:

print(r.data, end = ' ')

if d < r.data:

path(r.left, d)

else:

path(r.right, d)

else:

print(d)

def search(r, d):

"""return pointer to node that has data d """

if not r: # empty tree

return None

if r.data == d:

return r

else:

if d < r.data:

return search(r.left, d)

else:

return search(r.right, d)

def depth(r, d):

"""return depth of node data d """

if r.data == d:

return 0

else:

if d < r.data:

return depth(r.left, d)+1

else:

return depth(r.right, d)+1