# Lab Assignment 6

Name: Taskia Siddika

ID: 20301417

Section:03

# Task 1:

def LCA(root, x, y):

if root is None:

return None

if x < root.elem and y < root.elem:

return LCA(root.left, x, y)

if x > root.elem and y > root.elem:

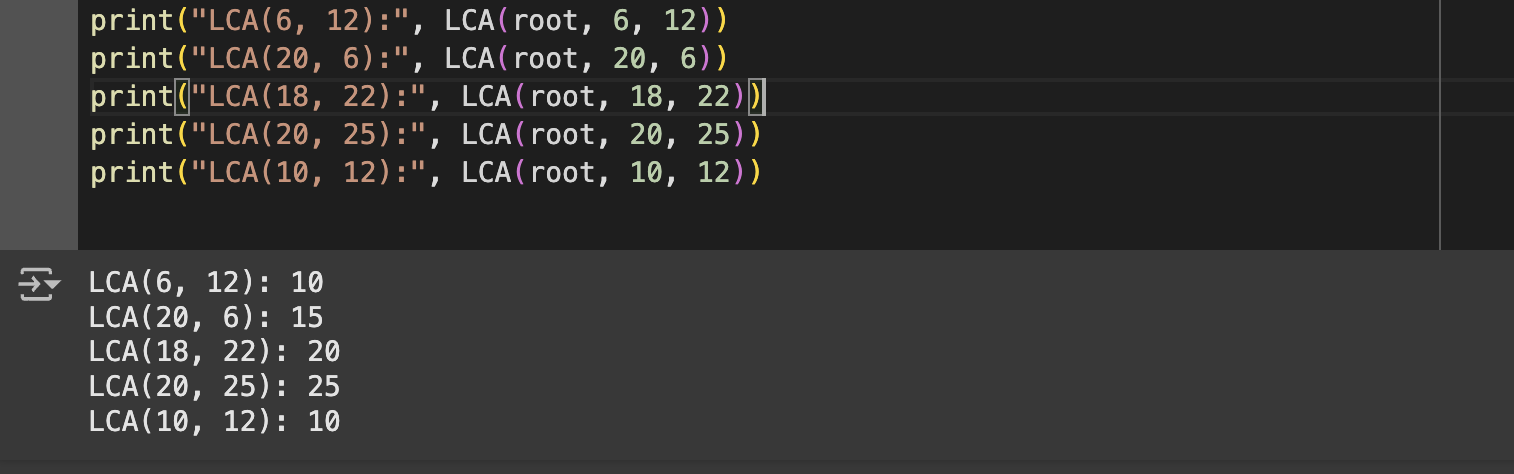
return LCA(root.right, x, y)

return root.elem

array\_representation = [None, 15, 10, 25, 8, 12, 20, 30, 6, 9, None, None, 18, 22]

root = tree\_construction(array\_representation)

**# Output Screenshot**

****

# 

# 

# 

# 

# 

# Task 2:

def find\_Path(root, key):

if root is None:

return "No Path Found"

if root.elem == key:

return [root.elem]

if key < root.elem:

left\_path = find\_Path(root.left, key)

if left\_path != "No Path Found":

return [root.elem] + left\_path

if key > root.elem:

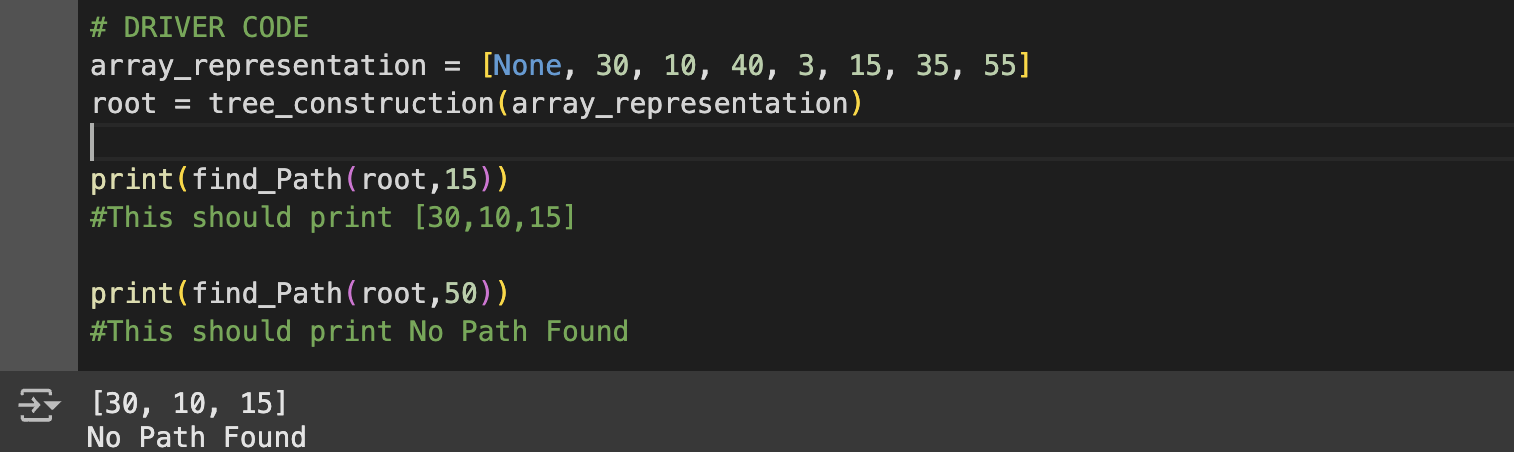
right\_path = find\_Path(root.right, key)

if right\_path != "No Path Found":

return [root.elem] + right\_path

return "No Path Found"

**# Output Screenshot**

****

TASK 3:

def sum\_of\_leaves(root, total):

if root is None:

return total

if root.left is None and root.right is None:

total += root.elem

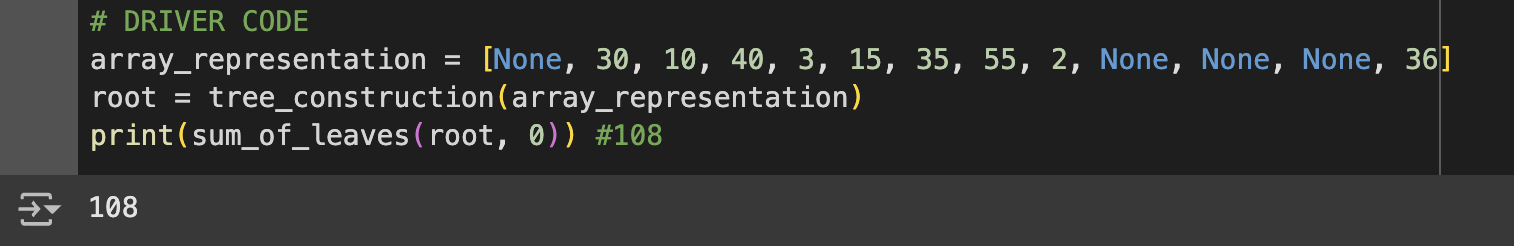
return total

total = sum\_of\_leaves(root.left, total)

total = sum\_of\_leaves(root.right, total)

return total

**# Output Screenshot**



TASK 4:

def inorder\_predecessor(root, x):

predecessor = None

for y in iter(int, 1):

if root is None:

break

if x.elem <= root.elem:

root = root.left

else:

predecessor = root

root = root.right

return predecessor

#output screenshot

