Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam - 603 110
(An Autonomous Institution, Affiliated to Anna University, Chennai)

UCS2403: DESIGN & ANALYSIS OF ALGORITHMS

Assignment 5

1. (a) First, find the k^{th} smallest element in an unsorted list using insertion sort

   (b) Next, find the element by modifying the divide-and-conquer algorithm of Quicksort

   (c) Compare the time complexity of both the algorithms

2. Consider the code given below that has to find the sum of the values in the nodes of a binary tree.

```
# Code to populate a tree starts here
import random
class TreeNode:
    def __init__(self):
        self.data = 0
        self.left = None
        self.right = None

    def insert(self, data):
        if data < self.data:
            if self.left == None:
                tempNode = TreeNode()
                self.left = tempNode
                self.left.data = data
            else:
                self.left.insert(data)
        elif data > self.data:
            if self.right == None:
                tempNode = TreeNode()
                self.right = tempNode
                self.right.data = data
            else:
                self.right.insert(data)

    def traverseInOrder(self):
        if self.left != None:
            self.left.traverseInOrder()
        print(self.data, end=’ ’)
        if self.right != None:
            self.right.traverseInOrder()

def createRoot():
    i = random.randint(0, 10)
    rootNode = TreeNode()
    rootNode.data = i
    return rootNode

def createTree():
    rootNode = createRoot()
    numNodes = random.randint(1, 10)
    currentNode = rootNode
    j = 0
    L = []
    while (j <= numNodes):
        newVal = random.randint(1,20)
        if newVal not in L:
            currentNode.insert(newVal)
            L.append(newVal)
        j+=1
    rootNode.traverseInOrder()
    return rootNode
# Code to populate the tree ends here

def getSum(node):
    if node == None:
        return 0
    else:
        leftSum = getSum(node.left)
        rightSum = getSum(node.right)
        return leftSum + rightSum

rootNode = createTree()
print("Sum = ",getSum(rootNode))
```

(a) The code is known to have some bugs. Modify the given program to correctly find the sum.

(b) Use Hypothesis to find counterexamples to show that the given code has errors.

(c) Please note that the number of nodes in the tree and the value in each node are generated randomly.