程式作業四

107502540陳皇宇

**資料結構**：

Define Node Type have four property

Node.name (word)

Node.value (freq)

Node.left

Node.right

NodeList and TreeNode to save Nodes

**演算法**：

先把input create 成 Node ，直接append到NodeList ，TreeNode 由小到大insert

接下來透過whlie 把tree 建好

最後透過遞迴找出所有code

**時間複雜度**：

Append input : O(N)

Create Tree:O(N-1)

Search Code:

T(N) = 2T() +1

O() > O(1) -> by master theorem T(N) = O(n)

**Time complexity O(N+N+N-1) = O(N)**

**Pseudo code**：

**function** insertNode(NodeList, newnode):

if len(NodeList) == 0:

NodeList.append(newnode)

return

for index, n in enumerate(NodeList):

if n.value < newnode.value:

NodeList.insert(index, newnode)

break

if index == len(NodeList)-1:

NodeList.append(newnode)

Break

**function** searchcode(node, code):

if node.name != ""

Codelist[node.name] = code

return

searchcode(node.left, code+"0")

searchcode(node.right, code+"1")

return

Codelist = {}

NodeList = []

TreeNode = []

for i from 0 to Nnum

name, value <- input

newnode = Node(name=name, value=value)

NodeList.append(newnode)

if len(TreeNode) == 0

TreeNode.append(newnode)

else

insertNode(TreeNode, newnode)

while(len(TreeNode) != 1):

minnode1 = TreeNode.pop()

minnode2 = TreeNode.pop()

newnode = Node(value=minnode1.value + minnode2.value, left=minnode1, right=minnode2)

insertNode(TreeNode, newnode)

searchcode(TreeNode[0], "")

for n in NodeList

print(n.name, " ", Codelist[n.name])