Assignment 3 Programming Questions

1)

Algorithm Clone(v)

v is the node of the Tree.

Input: v the node of the tree

Output: A copy of the Tree

Parent <-- Node(v.getElement, v.getParent, v.getLeft, v.getRight)

Rightnode < -- (Node)v

Leftnode <-- (Node)v

if Parent.getLeft **not** = **null**

LeftNode <-- clone(left(v)

Left <--v.LeftNode // Left is the left node of the upper node.

Parent <-- Left.Parent

end if

if Parent.getRight **not** = **null**

RightNode <-- clone(right(v)

Right <--v.RightNode // Left is the left node of the upper node.

Parent <-- Right.Parent

end if

return Parent

Algorithm iterator(v, inorder)

v is the node , inorder is an arrayList

Input: the node v, An array list called inorder

Output: Void

if left of v **not =** null

iterator(left of v , inorder) // this is a recursive call

inorder <-- add v.getElement

//here we add the element into the list when the right and left are null.

if right of v **not =** null

iterator(right of v , inorder) // this is a recursive call

4)

Structure is an array vs linked list

The complexity changes by using an ArrayList rather than a Linked list, because the array List has a faster time complexity. For the sake of this assignment, we are interested in having a fast time complexity since we know the size of the Child collection list will only have 2 children, And the number of nodes for the inorder list will be finite. Therefore the space complexity will not be that big, so it is safe to use arraylists instead of linked lists.

5)

Linked lists are capable of growing in size without having to create a new Array. It is faster to add nodes using linked list, and has good space complexity but worst time complexity.

ArrayLists are faster to access, but are slower to add new nodes to. Therefore arraylists have better time complexity but worst space complexity.