**ExtraExDS\_4Trees – Exercise 4**

**Specification**

class tree[node]

genres tree, node, label, bool, int

operations

parent: node tree 🡪 node

left\_mostchild: node tree 🡪 node

right\_sibling: node tree 🡪 node

label: node tree 🡪 label

create: tree tree label -> tree

root: tree 🡪 node

makenull: tree 🡪 tree

is\_masterTree: tree node 🡪 bool

get\_numberChildren: tree node 🡪 int

has\_moreChildren: tree 🡪 node

endspec

**Implementation**

node = record

element : label

leftmostchild: ^node

rightsibling: ^node

parent: ^node

endrecord

label: elementtype

class Tree

private root: ^node

public ^node parent(n: node)

public ^node left\_mostchild(n: node)

public ^node right\_sibling(n: node)

public label label(n: node)

public void create(leftTree, rightTree: ^node, l: label)

public ^node root()

public void makenull()

public bool is\_masterTree(n: ^node)

private int get\_numberChildren(n: ^node)

public ^node has\_moreChildren()

endclass

public bool Tree:: is\_masterTree(n: ^node)

counter\_Children: int

temp: ^node

verify

counter:= 0

if (n == null)

return true

counter\_Children = get\_numberChildren(n^.leftmost\_child)

if (n^.element == counter\_Children)

verify:= is\_masterTree(n^.leftmost\_child)

if (verify)

temp := n^.leftmost\_Child^.right\_sibling

while temp != null

verify:= is\_masterTree(temp)

if(verify)

temp:= temp^.right\_sibling

else

return false

endif

endwhile

return true

else

return false

endif

else

return false

endif

endmethod

private int Tree:: get\_numberChildren(n: ^node)

counter : int

counter := 0

if (n == null)

return counter

else

counter:= get\_numberChildren(n^.right\_sibling)

counter := counter + 1

return counter

endif

endmethod

public ^node Tree:: has\_moreChildren(temp1, temp2: ^node) {temp1 is the new node, temp2 the

greater\_node, n: ^node greater until that moment}

if (temp1 == null)

return null

endif

if(temp1^.element > temp2^.element || temp2 == null) {temp2 will be null when temp1 is

greater\_node:= temp1 the root}

endif

greater\_node:= has\_moreChildren(temp1^.leftmost\_child, greater\_node)

n:= temp1^.leftmost\_child

while n != null

greater\_node:= has\_moreChildren(n, greater\_node)

n:= n^.right\_sibling

endwhile

return greater\_node

endmethod