Name: Galbenu Horia (ghie1544)

Group: 922

Lab code: L2

Lab delivery date: 19/12/2014

Problem number and statement: Problem 9

Return the level of a node in a tree of type (2). The level of the root is 0.

Formal descriptions:

- Mathematical models:

left-subtree(tree):
- nil, tree null
- (car(cdr(tree))), otherwise

right-subtree(tree):
- nil, tree null
- (car(cdr(cdr(tree)))), otherwise

get-level(tree node ct):
- -1, tree null
- ct, if (car(tree)) = node
- (get-level (left-subtree tree) node (+ 1 ct), if !=0
- (get-level (right-subtree tree) node (+ 1 ct), else

- Meaning of function parameters:
  tree – n-ary three represented as a list.
  node – node we'll search for and get its level
  counter – used to compute the level of the node we're looking for

Source code:

```
( defun left-subtree( tree )
  ( cond( ( null tree ) nil )
    ( t ( car ( cdr tree ) ) ) ))
```

```
( defun right-subtree( tree )
  ( cond( ( null tree ) nil )
    ( t ( car ( cdr ( cdr tree )  ) ) ) ))

( defun get-level( tree node counter )
  ( cond ( ( null tree ) -1 )
    ( ( eql (car tree) node )  counter )
    ( t  ( setq func ( get-level ( left-subtree tree ) node ( + 1 counter ) ) )
         ( cond  ( ( not ( eql func -1 ) ) func )
           ( t ( get-level ( right-subtree tree ) node ( + 1 counter ) ) ) ) )))
```

Running examples:

(get-level '(a (b) (c (d) (e))) 'd '0)-> (get-level (c (d) (e)) ‘d ‘1) -> (get-level (d) ‘d ‘2)->

-> 2