Notes lecture 9

1. **Functions with destructive access to the contents of lists:**

* **RPLACA subr 2 (l e): l** = replace CAR of list l with evaluated value of expression e
* **RPLACD subr 2 (l e): l** = replace CDR of list l with evaluated value of expression e
* **NCONC lsubr 1- (f l1 l2 ... ln): l** = APPEND, but modifies the first list

A black text on a white background

Description automatically generated

* **REMOVE subr 2 (e l): l** = copy of l with all elems EQL to e removed (superficial lvl)
* **DELETE subr 2 (e l): l** = same, but alters list l (EXCEPT FIRST ELEM)
* **SUBST subr 3 (e1 e2 e3): e**
  + copy of e3 in which all occurrences, at any level, of the expression e2 have been replaced by e1
* **REMOVE-IF nsubr 2 (e l): l** 
  + copy of l in which all elems that satisfy condition e are removed
* **DELETE-IF** = destructive variant

1. **Binary trees**

* V1: (root list-subtree-left list-subtree-right)
* V2: (root nr-subtrees

representation of subtree-1

representation of subtree-2

…)

1. **Probleme:**
   1. **Inorder traversal of tree in V2 representation:**
   * Determine the left/right subtree of a tree
     + Left = from 3rd position until
     + Right = from where until the end
   * Inorder(tree) = Inorder(left(tree)) + root + Inorder(right(tree))
   1. **Tree sorting of a linear list:**
   * create\_tree(e, l1l2l3) = {

(e), if l empty

l1 + create\_tree(e, l2) + l3, if e <= l1

l1 + l2 + create\_tree(e, l3), otherwise

}

* + Do an in-order traversal of the created tree