Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Peter Lammich

SS 2017
9. 6. 2017

Functional Data Structures

Exercise Sheet 7

Exercise 7.1 Round wrt. Binary Search Tree

The distance between two integers x and y is $|x - y|$.

1. Define a function *round* :: *int tree* ⇒ *int* ⇒ *int option*, such that *round t x* returns an element of a **binary search tree** t with minimum distance to x, and *None* if and only if t is empty.

   Define your function such that it does no unnecessary recursions into branches of the tree that are known to not contain a minimum distance element.
2. Specify and prove that your function is correct. Note: You are required to phrase the correctness properties yourself!

   Hint: Specify 3 properties:

   - None is returned only for the empty tree.
   - Only elements of the tree are returned.
   - The returned element has minimum distance.
3. Estimate the time of your round function to be linear in the height of the tree

fun *round* :: *"int tree ⇒ int ⇒ int option"*
fun *t_round* :: *"int tree ⇒ int ⇒ nat"*

Homework 7 Cost for *remdups*

Submission until Friday, 16. 6. 2017, 11:59am.

The following function removes all duplicates from a list. It uses the auxiliary function *member* to determine whether an element is contained in a list.

fun *member* :: *"'a ⇒ 'a list ⇒ bool"* **where**
  "member x [] ⟷ False"
| *"member x (y#ys) ⟷ (if x=y then True else member x ys)"*

fun *rem_dups* :: *"'a list ⇒ 'a list"* **where**
  "rem_dups [] = []" |

"rem_dups (x # xs) = (if member x xs then rem_dups xs else x # rem_dups xs)"

Show that this function is equal to the HOL standard function *remdups*

lemma *rem_dups_correct*: *"rem_dups xs = remdups xs"*

Define the timing functions for *member* and *rem_dups*, as described on the slides:

fun *t_member* :: *"'a ⇒ 'a list ⇒ nat"*
fun *t_rem_dups* :: *"'a list ⇒ nat"*

Estimate *t_rem_dups xs* to be quadratic in the length of *xs*. Hint: The estimate (*length xs + 1*)^2 should work.