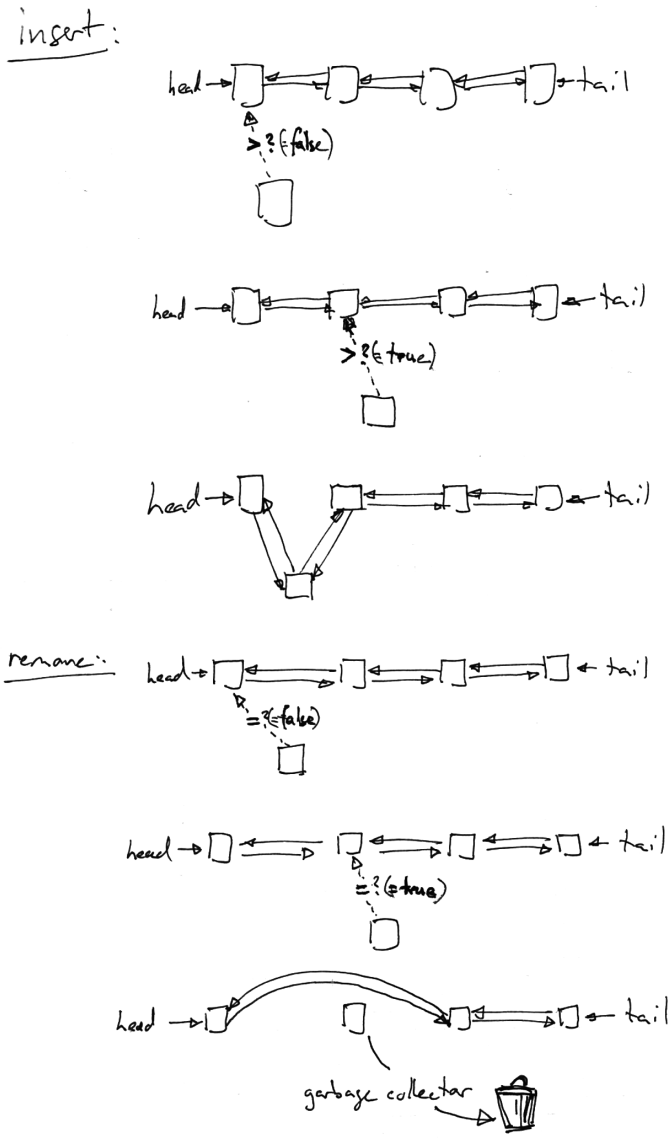
Lösungen Blatt 4

Aufgaben 1a und 1c

Wir haben die Aufgabe mit Test-Driven Development gelöst und Teilaufgabe (c) somit kontinuierlich und jeweils vor der Implementation jedes Codeteiles durchgeführt. Der Code ist in brunnerrabe/blatt4/aufgabe1/.

Die Versionshistorie und damit die Vorgehensweise ist auf GitHub online einsehbar.[1]

Aufgabe 1b



1 https://github.com/felixrabe/brunnerrabe_blatt4/commits/master. Das Repository ist auch in der eingereichten ZIP-Datei enthalten (.git/) und per Git (http://git-scm.com/) zugänglich.

Aufgabe 2

Diese Aufgabe haben wir basierend auf Aufgabe 1 gelöst. Die Unterschiede sind auf GitHub klar ersichtlich.[2] Der Code ist in brunnerrabe/blatt4/aufgabe2/.

Aufgabe 3

Wurde bereits im Unterricht gelöst.

Aufgabe 4

```
type Bool
operators
    true : -> Bool
    false : -> Bool

type Nat
operators
    0 : -> Nat
    suc : Nat -> Nat
    add : Nat × Nat -> Nat
axioms ∀i,j : Nat
    add (i, 0) = i
    add (i, suc (j)) = suc (add (i, j))

type X
import Bool
operators
    _ ≠ _ : -> Bool
    _ > _ : -> Bool
```

2 https://github.com/felixrabe/brunnerrabe_blatt4/compare/aufgabe1...aufgabe2.

```
type SortSeq(X)
import Bool, Nat
operators
    mt_SortSeq : -> SortSeq
    ( cons : X × SortSeq -> SortSeq )
    insert : X × SortSeq -> SortSeq
    del : X × SortSeq -> SortSeq
    contains : X × SortSeq -> Bool
    isEmpty : SortSeq -> Bool
    size : SortSeq -> Nat
    min : SortSeq -> X
    max : SortSeq -> X
    clear : SortSeq -> SortSeq
axioms ∀s : SortSeq(X), ∀x,y : X ∧ x ≠ y
    insert (x, mt_SortSeq) = cons (x, mt_SortSeq)
    insert (x, cons (x, s)) = cons (x, cons (x, s))
    insert (x, cons (y, s)) =
        if x > y then cons (y, insert (x, s))
                 else cons (x, cons (y, s))    /* x < y */
    del (x, mt_SortSeq) = ⊥ /* oder: del (x, mt_SortSeq) = mt_SortSeq */
    del (x, cons (x, s)) = s
    del (x, cons (y, s)) = cons (y, del (x, s))
    contains (x, mt_SortSeq) = false
    contains (x, cons (x, s)) = true
    contains (x, cons (y, s)) = contains (x, s)
    isEmpty (mt_SortSeq) = true
    isEmpty (cons (x, s)) = false
    size (mt_SortSeq) = 0
    size (cons (x, s)) = suc (size (s))
    min (mt_SortSeq) = ⊥
    min (cons (x, s)) = x
    max (mt_SortSeq) = ⊥
    max (cons (x, mt_SortSeq)) = x
    max (cons (x, s)) = max (s)
    clear (s) = mt_SortSeq
```