(circular list) Design axioms for a circular list. There should be operations to create an empty list, to move along one position in the list (the first item comes after the last, in circular fashion), to insert an item at the current position, to delete the current item, and to give the current item.

§ Let the type of items in the list be $X$. I introduce two implementer's variables. Let $L$ be a list of items of type $X$, and let $i$ be an index of $L$. The theory introduces six names.

- **new** a program that makes an empty list
- **mvr** a program that moves right one position in the list
- **mvl** a program that moves left one position in the list
- **ins** a program with parameter $x$ that inserts $x$ into the list at the current position
- **del** a program that deletes the item at the current position
- **val** the value of the item at the current position in the list

Using $\text{nat}$ for $X$, here is an example list and current position in the list.

$L = [5; 7; 1; 8; 6]$

$mvr = \text{if } #L = 0 \text{ or } i = #L - 1 \text{ then } i := 0 \text{ else } i := i + 1 \text{ fi}$

$mvl = \text{if } #L = 0 \text{ then } i := 0 \text{ else if } i = 0 \text{ then } i := #L - 1 \text{ else } i := i - 1 \text{ fi fi}$

$ins = \langle x : X \cdot L := L[0;..i] ;; [x] ;; L[i;..#L] \rangle$

$del = L := L[0;..i] ;; L[i+1;..#L]$

$val = L i$

If the list is empty, then $\text{del}$ and $\text{val}$ are undefined. We could strengthen the theory by defining them. Maybe we should add

- **emp** a binary saying whether the list is empty

and define it as

$\text{emp} = #L = 0$

so the programmer can test whether the list is empty before using $\text{val}$ and $\text{del}$. If we do add it, we can simplify $\text{mvr}$ and $\text{mvl}$

$mvr = \text{if } i = #L - 1 \text{ then } i := 0 \text{ else } i := i + 1 \text{ fi}$

$mvl = \text{if } i = 0 \text{ then } i := #L - 1 \text{ else } i := i - 1 \text{ fi}$

because the programmer can test whether the list is empty before using $\text{mvr}$ and $\text{mvl}$.