MODULE *Op*

Model checking basic operations on strings (*i.e.*, list of characters).

EXTENDS $\mathit{Naturals}$, $\mathit{Sequences}$,
    $\mathit{AdditionalMathOperators}$, $\mathit{AdditionalSetOperators}$, $\mathit{AdditionalSequenceOperators}$

CONSTANTS Char set of characters allowed

$\mathit{List} \triangleq \mathit{Seq}(\mathit{Char})$ all possible lists/strings

The set of all operations.

$\mathit{Rd} \triangleq [\mathit{type} : \{\text{“Rd”}\}]$ a read specifies no arguments
$\mathit{Ins} \triangleq [\mathit{type} : \{\text{“Del”}\}, \mathit{pos} : \mathit{PosInt}]$ a deletion specifies a position, indexed from 1
$\mathit{Del} \triangleq [\mathit{type} : \{\text{“Ins”}\}, \mathit{pos} : \mathit{PosInt}, \mathit{ch} : \mathit{Char}, \mathit{pr} : \mathit{PosInt}]$ an insertion also specifies a character and a priority

$\mathit{Op} \triangleq \mathit{Ins} \cup \mathit{Del}$ Now we focus on “Ins” and “Del”.
$\mathit{Nop} \triangleq \mathit{PickNone}(\mathit{Op})$ *Nop*: an operation representing “doing nothing”

Some operations for test.

$\mathit{Del}1 \triangleq [\mathit{type} \mapsto \text{“Del”}, \mathit{pos} \mapsto 1]$
$\mathit{Del}2 \triangleq [\mathit{type} \mapsto \text{“Del”}, \mathit{pos} \mapsto 2]$
$\mathit{Del}3 \triangleq [\mathit{type} \mapsto \text{“Del”}, \mathit{pos} \mapsto 3]$
$\mathit{Del}4 \triangleq [\mathit{type} \mapsto \text{“Del”}, \mathit{pos} \mapsto 4]$
$\mathit{Ins}1 \triangleq [\mathit{type} \mapsto \text{“Ins”}, \mathit{pos} \mapsto 1, \mathit{ch} \mapsto \text{“a”}, \mathit{pr} \mapsto 1]$
$\mathit{Ins}2 \triangleq [\mathit{type} \mapsto \text{“Ins”}, \mathit{pos} \mapsto 2, \mathit{ch} \mapsto \text{“b”}, \mathit{pr} \mapsto 2]$
$\mathit{Ins}3 \triangleq [\mathit{type} \mapsto \text{“Ins”}, \mathit{pos} \mapsto 3, \mathit{ch} \mapsto \text{“c”}, \mathit{pr} \mapsto 3]$
$\mathit{Ops} \triangleq \langle \mathit{Ins}2, \mathit{Del}3, \mathit{Ins}1, \mathit{Del}2, \mathit{Ins}3, \mathit{Del}1 \rangle$

The “Apply” operator which applies an operation *op* on the list *l*.

Del: If *pos* > *Len*(*l*), the last element of *l* is deleted. This is realized by the *DeleteElement* operator.

Ins: If *pos* > *Len*(*l*), the new element is appended to *l*. This is realized by the *InsertElement* operator.

```
$\mathit{Apply}(\mathit{op}, l) \triangleq$ CASE $\mathit{op} = \mathit{Nop} \rightarrow l$
                    □ $\mathit{op.type} = \text{“Rd”} \rightarrow l$
                    □ $\mathit{op.type} = \text{“Del”} \rightarrow \mathit{DeleteElement}(l, \mathit{op.pos})$
                    □ $\mathit{op.type} = \text{“Ins”} \rightarrow \mathit{InsertElement}(l, \mathit{op.ch}, \mathit{op.pos})$
```

The “*ApplyOps*” operator which applies an operation sequence *ops* on the list *l*.

```
RECURSIVE $\mathit{ApplyOps}(\_, \_)$
$\mathit{ApplyOps}(\mathit{ops}, l) \triangleq$
   IF $\mathit{ops} = \langle \rangle$
    THEN $l$
    ELSE $\mathit{Apply}(\mathit{Last}(\mathit{ops}), \mathit{ApplyOps}(\mathit{AllButLast}(\mathit{ops}), l))$
```

Check whether an operation *op* is legal with respect to the list *l*.

```
IsLegalOp(op, l) ≜ CASE op.type = "Del" → op.pos ≤ Len(l)
                        □    op.type = "Ins"  → op.pos ≤ Len(l) + 1

Legalize an operation op with respect to the list l.
 LegalizeOp(op, l) ≜ CASE op.type = "Del" →  [op  EXCEPT !.pos = Min(@, Len(l))]
                          □op.type = "Ins" →  [op  EXCEPT !.pos = Min(@, Len(l) + 1)]

\ *  Modification History
\ * Last modified Sun Jul 08 10:11:21 CST 2018 by hengxin
\ *  Created Sat Jun 23 20:56:53 CST 2018 by hengxin
```