MODULE *AdditionalSequenceOperators*

Copyright: https://*github.com*/bringhurst/tlaplus/blob/master/*org.lamport.tla.toolbox.uitest*/farsite/*AdditionalSequenceOperat*

EXTENDS *FiniteSets*, *Sequences*, *TLC*, *AdditionalSetOperators*, *AdditionalFunctionOperators*

LOCAL INSTANCE *Naturals*

The TLA+ *Sequences* module defines the operators *Head* and *Tail* for retrieving the first element of a sequence and all-but-the-first elements of a sequence, respectively. This module provides four operators that slightly generalize the notions of *Head* and *Tail*:

First returns the first element of a sequence, equivalently to *Head*. Last returns the last element of a sequence. *AllButFirst* returns all-but-the-first elements of a sequence, equivalently to *Tail*.

AllButLast returns all-but-the-last elements of a sequence.

This module also provides several additional operators on sequences: *IsElementInSeq* is a predicate that is true when the specified value is an element of the specified sequence. *IsSequenceOfSetElements* is a predicate that is true when the specified sequence contains all and only elements of the specified set. *IsSortedSequenceOfSetElements* is a predicate that is true when the *IsSequenceOfSetElements* is true and the sequence is also sorted in increasing order. *DeleteElement* produces a sequence by deleting an indicated element from another sequence.

$$Prepend(s,\ e) \triangleq \langle e \rangle \circ s$$

$$First(seq) \triangleq seq[1]$$

$$Last(seq) \triangleq seq[Len(seq)]$$

$$AllButFirst(seq) \triangleq [i \in 1\ ..\ (Len(seq) - 1) \mapsto seq[(i + 1)]]$$

$$AllButLast(seq) \triangleq [i \in 1\ ..\ (Len(seq) - 1) \mapsto seq[i]]$$

$$\begin{aligned} &DoesSeqPrefixSeq(seq1,\ seq2) \triangleq \\ &\quad \land\ Len(seq1) \leq Len(seq2) \\ &\quad \land\ (\forall\ i \in 1\ ..\ Len(seq1) : seq1[i] = seq2[i]) \end{aligned}$$

$$\begin{aligned} &DoesSeqProperlyPrefixSeq(seq1,\ seq2) \triangleq \\ &\quad \land\ Len(seq1) < Len(seq2) \\ &\quad \land\ (\forall\ i \in 1\ ..\ Len(seq1) : seq1[i] = seq2[i]) \end{aligned}$$

$$IsElementInSeq(el,\ seq) \triangleq \exists\ i \in \text{DOMAIN}\ seq : seq[i] = el$$

$$\begin{aligned} &IsSequenceOfSetElements(seq,\ set) \triangleq \\ &\quad \land\ Len(seq) = Cardinality(set) \\ &\quad \land\ (\forall\ el \in set : IsElementInSeq(el,\ seq)) \end{aligned}$$

$$\begin{aligned} &IsSortedSequenceOfSetElements(seq,\ set) \triangleq \\ &\quad \land\ IsSequenceOfSetElements(seq,\ set) \\ &\quad \land\ (\forall\ i \in \text{DOMAIN}\ seq,\ j \in \text{DOMAIN}\ seq : i < j \Rightarrow seq[i] < seq[j]) \end{aligned}$$

$$DeleteElement(seq,\ index) \triangleq$$

61 $[i \in 1 \,..\, (Len(seq) - 1) \mapsto \text{IF } i < index \text{ THEN } seq[i] \text{ ELSE } seq[(i+1)]]$

It requires that $index \geq 1$.

If $index > Len(seq) + 1$, then it appends the element to seq.

(ADDED by hengxin; *July* 04, 2018)

70 $InsertElement(seq, elem, index) \triangleq$
71 $[i \in 1 \,..\, (Len(seq) + 1) \mapsto \text{IF } i < index$
72 THEN IF $i = (Len(seq) + 1)$
73 THEN $elem$
74 ELSE $seq[i]$
75 ELSE IF $i = index$
76 THEN $elem$
77 ELSE $seq[(i-1)]]$ $i > index$

79 $IsSorted2Partition(n, seq1, seq2) \triangleq$
80 $\land\ seq1 \in Seq(1 \,..\, n)$
81 $\land\ seq2 \in Seq(1 \,..\, n)$
82 $\land\ n = Len(seq1) + Len(seq2)$
83 $\land\ (\forall\, i \in \text{DOMAIN } seq1, j \in \text{DOMAIN } seq1 : i < j \Rightarrow seq1[i] < seq1[j])$
84 $\land\ (\forall\, i \in \text{DOMAIN } seq2, j \in \text{DOMAIN } seq2 : i < j \Rightarrow seq2[i] < seq2[j])$
85 $\land\ (\forall\, i \in \text{DOMAIN } seq1, j \in \text{DOMAIN } seq2 : seq1[i] \neq seq2[j])$

87 $IsSequenceInterleaving(seq, subSeq1, subSeq2, indSeq1, indSeq2) \triangleq$
88 $\land\ indSeq1 \in Seq(Nat)$
89 $\land\ indSeq2 \in Seq(Nat)$
90 $\land\ IsSorted2Partition(Len(seq), indSeq1, indSeq2)$
91 $\land\ Len(indSeq1) = Len(subSeq1)$
92 $\land\ Len(indSeq2) = Len(subSeq2)$
93 $\land\ (\forall\, i \in \text{DOMAIN } indSeq1 : seq[(indSeq1[i])] = subSeq1[i])$
94 $\land\ (\forall\, i \in \text{DOMAIN } indSeq2 : seq[(indSeq2[i])] = subSeq2[i])$

Sequences up to length n, including the empty sequence $\langle\rangle$.

Copyright: https://*www.learntla.com*/libraries/sequences/

101 $SeqMaxLen(S, n) \triangleq \text{UNION } \{[1 \,..\, m \to S] : m \in 0 \,..\, n\}$

Map on a sequence.

Copyright: https://*www.learntla.com*/libraries/sequences/

108 $SeqMap(Op(\_), seq) \triangleq [x \in \text{DOMAIN } seq \mapsto Op(seq[x])]$

110 $PermsWithin(S) \triangleq \{s \in \text{UNION } \{[1 \,..\, m \to S] : m \in 0 \,..\, Cardinality(S)\} : Cardinality(Range(s)) = Cardina$

All possible permutations generated based on sequence T.

Copyright: https://*learntla.com*/tla/functions/

117 $PermutationKey(n) \triangleq \{key \in [1 \,..\, n \to 1 \,..\, n] : Range(key) = 1 \,..\, n\}$

```
118 PermutationsOf(T) ≜ {[x ∈ 1 .. Len(T) ↦ T[P[x]]] : P ∈ PermutationKey(Len(T))}
```

Get the index of the first occurrence of *elem* in *seq*.

Precondition: $elem \in SeqImage(seq)$.

ADDED by hengxin; *Aug.* 12, 2018

```
126 RECURSIVE FirstIndexOfElement(_, _)
127 FirstIndexOfElement(seq, elem) ≜
128     IF Head(seq) = elem
129      THEN 1
130      ELSE 1 + FirstIndexOfElement(Tail(seq), elem)
```

Check if two sequences are compatible.

Precondition: No duplication in each individual sequence.

Two sequences are compatible if and only if for any two common elements in both sequences, the relative order of them in the two sequences are the same.

ADDED by hengxin; *Aug.* 12, 2018

```
143 Compatible(seq1, seq2) ≜
144     ∨ seq1 = seq2
145     ∨ LET commonElements ≜ Range(seq1) ∩ Range(seq2)
146       IN  ∀ e1, e2 ∈ commonElements :
147             ∨ e1 = e2
148             ∨ FirstIndexOfElement(seq1, e1) < FirstIndexOfElement(seq1, e2)
149               ≡ FirstIndexOfElement(seq2, e1) < FirstIndexOfElement(seq2, e2)
```

The length of the longest common subsequence of two sequences *seq*1 and *seq*2.

ADDED by hengxin; *Aug.* 12, 2018

```
156 RECURSIVE LCS(_, _)
157 LCS(seq1, seq2) ≜
158       IF seq1 = ⟨⟩ ∨ seq2 = ⟨⟩
159        THEN 0
160        ELSE IF Last(seq1) = Last(seq2)
161                THEN 1 + LCS(AllButLast(seq1), AllButLast(seq2))
162                ELSE MaxOfSet({LCS(AllButLast(seq1), seq2), LCS(seq1, AllButLast(seq2))})

164 LCSCompatible(seq1, seq2) ≜
165     Compatible(seq1, seq2) ≡ LCS(seq1, seq2) = Cardinality(Range(seq1) ∩ Range(seq2))

167 LCSCompatibleTest(S) ≜
168     ∀ seq1, seq2 ∈ PermsWithin(S) : LCSCompatible(seq1, seq2)
169 ====================================================================
```

* Modification History
* *Last* modified *Mon Sep* 03 20:26:21 *CST* 2018 by *hengxin*
* Created *Tue Jul* 03 15:21:02 *CST* 2018 by *hengxin*