1 ———————————— MODULE *AdditionalSequenceOperators* ————————————

4

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

8 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.

32 $Prepend(s, e) \triangleq \langle e \rangle \circ s$

34 $First(seq) \triangleq seq[1]$

36 $Last(seq) \triangleq seq[Len(seq)]$

38 $AllButFirst(seq) \triangleq [i \in 1 .. (Len(seq) - 1) \mapsto seq[(i + 1)]]$

40 $AllButLast(seq) \triangleq [i \in 1 .. (Len(seq) - 1) \mapsto seq[i]]$

42 $DoesSeqPrefixSeq(seq1, seq2) \triangleq$
43 $\quad \land Len(seq1) \leq Len(seq2)$
44 $\quad \land (\forall i \in 1 .. Len(seq1) : seq1[i] = seq2[i])$

46 $DoesSeqProperlyPrefixSeq(seq1, seq2) \triangleq$
47 $\quad \land Len(seq1) < Len(seq2)$
48 $\quad \land (\forall i \in 1 .. Len(seq1) : seq1[i] = seq2[i])$

50 $IsElementInSeq(el, seq) \triangleq \exists i \in \text{DOMAIN } seq : seq[i] = el$

52 $IsSequenceOfSetElements(seq, set) \triangleq$
53 $\quad \land Len(seq) = Cardinality(set)$
54 $\quad \land (\forall el \in set : IsElementInSeq(el, seq))$

56 $IsSortedSequenceOfSetElements(seq, set) \triangleq$
57 $\quad \land IsSequenceOfSetElements(seq, set)$
58 $\quad \land (\forall i \in \text{DOMAIN } seq, j \in \text{DOMAIN } seq : i < j \Rightarrow seq[i] < seq[j])$

60 $DeleteElement(seq, index) \triangleq$

1

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

Retain only the elements in R in their original order in seq.

66   RECURSIVE $Retain(\_, \_)$
67   $Retain(seq, R) \triangleq$
68   IF $seq = \langle \rangle$
69   THEN $\langle \rangle$
70   ELSE LET $h \triangleq Head(seq)$
71   IN   IF $h \in R$
72   THEN $\langle h \rangle \circ Retain(Tail(seq), R)$
73   ELSE $Retain(Tail(seq), R)$

It requires that $index \geq 1$.

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

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

81   $InsertElement(seq, elem, index) \triangleq$
82    $[i \in 1 \, .. \, (Len(seq) + 1) \mapsto \text{IF } i < index$
83   THEN IF $i = (Len(seq) + 1)$
84   THEN $elem$
85   ELSE $seq[i]$
86   ELSE IF $i = index$
87   THEN $elem$
88   ELSE $seq[(i-1)]]]$ $i > index$

90   $IsSorted2Partition(n, seq1, seq2) \triangleq$
91     $\land seq1 \in Seq(1 \, .. \, n)$
92     $\land seq2 \in Seq(1 \, .. \, n)$
93     $\land n = Len(seq1) + Len(seq2)$
94     $\land (\forall i \in \text{DOMAIN } seq1, j \in \text{DOMAIN } seq1 : i < j \Rightarrow seq1[i] < seq1[j])$
95     $\land (\forall i \in \text{DOMAIN } seq2, j \in \text{DOMAIN } seq2 : i < j \Rightarrow seq2[i] < seq2[j])$
96     $\land (\forall i \in \text{DOMAIN } seq1, j \in \text{DOMAIN } seq2 : seq1[i] \neq seq2[j])$

98   $IsSequenceInterleaving(seq, subSeq1, subSeq2, indSeq1, indSeq2) \triangleq$
99     $\land indSeq1 \in Seq(Nat)$
100    $\land indSeq2 \in Seq(Nat)$
101    $\land IsSorted2Partition(Len(seq), indSeq1, indSeq2)$
102    $\land Len(indSeq1) = Len(subSeq1)$
103    $\land Len(indSeq2) = Len(subSeq2)$
104    $\land (\forall i \in \text{DOMAIN } indSeq1 : seq[(indSeq1[i])] = subSeq1[i])$
105    $\land (\forall i \in \text{DOMAIN } indSeq2 : seq[(indSeq2[i])] = subSeq2[i])$

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

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

112 $SeqMaxLen(S, n) \triangleq$ UNION $\{[1 \, .. \, m \to S] : m \in 0 \, .. \, n\}$

119 $SeqMap(Op(\_),\ seq)\ \triangleq\ [x \in \text{DOMAIN } seq \mapsto Op(seq[x])]$

121 $PermsWithin(S)\ \triangleq\ \{s \in \text{UNION }\{[1 \mathinner{\ldotp\ldotp} m \to S] : m \in 0 \mathinner{\ldotp\ldotp} Cardinality(S)\} : Cardinality(Range(s)) = Cardin$

128 $PermutationKey(n)\ \triangleq\ \{key \in [1 \mathinner{\ldotp\ldotp} n \to 1 \mathinner{\ldotp\ldotp} n] : Range(key) = 1 \mathinner{\ldotp\ldotp} n\}$
129 $PermutationsOf(T)\ \triangleq\ \{[x\ \in 1 \mathinner{\ldotp\ldotp} Len(T) \mapsto T[P[x]]] : P \in PermutationKey(Len(T))\}$

137 RECURSIVE $FirstIndexOfElement(\_,\ \_)$
138 $FirstIndexOfElement(seq,\ elem)\ \triangleq$
139   IF $Head(seq) = elem$
140   THEN 1
141   ELSE $1 + FirstIndexOfElement(Tail(seq),\ elem)$

143 RECURSIVE $FirstIndexOfElementSafe(\_,\ \_)$
144 $FirstIndexOfElementSafe(seq,\ elem)\ \triangleq$
145   LET RECURSIVE $FirstIndexOfElementSafeHelper(\_,\ \_,\ \_)$
146          $FirstIndexOfElementSafeHelper(seqh,\ elemh,\ fail)\ \triangleq$
147   IF $seqh = \langle\rangle$
148   THEN $0 - fail$
149   ELSE IF $Head(seqh) = elemh$
150   THEN 1
151   ELSE $1 + FirstIndexOfElementSafeHelper(Tail(seqh),\ elemh,\ fail + 1)$
152   IN $FirstIndexOfElementSafeHelper(seq,\ elem,\ 0)$

164 $Compatible(seq1,\ seq2)\ \triangleq$
165      $\vee\ seq1 = seq2$
166      $\vee$ LET $commonElements\ \triangleq\ Range(seq1) \cap Range(seq2)$
167   IN $\forall\ e1,\ e2 \in commonElements :$
168              $\vee\ e1 = e2$
169              $\vee\ FirstIndexOfElement(seq1,\ e1) < FirstIndexOfElement(seq1,\ e2)$
170                  $\equiv FirstIndexOfElement(seq2,\ e1) < FirstIndexOfElement(seq2,\ e2)$

The length of the longest common subsequence of two sequences *seq1* and *seq2*.

ADDED by hengxin; *Aug.* 12, 2018

177 RECURSIVE $LCS(\_, \_)$
178 $LCS(seq1, seq2) \triangleq$
179   IF $seq1 = \langle \rangle \vee seq2 = \langle \rangle$
180   THEN 0
181   ELSE IF $Last(seq1) = Last(seq2)$
182   THEN $1 + LCS(AllButLast(seq1), AllButLast(seq2))$
183   ELSE $MaxOfSet(\{LCS(AllButLast(seq1), seq2), LCS(seq1, AllButLast(seq2))\})$

185 $LCSCompatible(seq1, seq2) \triangleq$
186      $Compatible(seq1, seq2) \equiv LCS(seq1, seq2) = Cardinality(Range(seq1) \cap Range(seq2))$

188 $LCSCompatibleTest(S) \triangleq$
189      $\forall\, seq1,\, seq2 \in PermsWithin(S) : LCSCompatible(seq1, seq2)$
190 └──

\ * Modification History
\ * *Last* modified *Mon Nov* 05 21:06:18 *CST* 2018 by *hengxin*
\ * Created *Tue Jul* 03 15:21:02 *CST* 2018 by *hengxin*

4