```
1 ---------- MODULE SequenceUtils ----------
```

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

```
6 EXTENDS FiniteSets, Sequences, SetUtils, FunctionUtils
7 LOCAL INSTANCE Naturals
```

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.

```
18 Prepend(s, e) ≜ ⟨e⟩ ∘ s

20 First(seq) ≜ seq[1]

22 Last(seq) ≜ seq[Len(seq)]

24 AllButFirst(seq) ≜ [i ∈ 1 .. (Len(seq) − 1) ↦ seq[(i + 1)]]

26 AllButLast(seq) ≜ [i ∈ 1 .. (Len(seq) − 1) ↦ seq[i]]

28 DoesSeqPrefixSeq(seq1, seq2) ≜
29   ∧ Len(seq1) ≤ Len(seq2)
30   ∧ (∀ i ∈ 1 .. Len(seq1) : seq1[i] = seq2[i])

32 DoesSeqProperlyPrefixSeq(seq1, seq2) ≜
33   ∧ Len(seq1) < Len(seq2)
34   ∧ (∀ i ∈ 1 .. Len(seq1) : seq1[i] = seq2[i])

36 IsElementInSeq(el, seq) ≜ ∃ i ∈ DOMAIN seq : seq[i] = el

38 IsSequenceOfSetElements(seq, set) ≜
39   ∧ Len(seq) = Cardinality(set)
40   ∧ (∀ el ∈ set : IsElementInSeq(el, seq))

42 IsSortedSequenceOfSetElements(seq, set) ≜
43   ∧ IsSequenceOfSetElements(seq, set)
44   ∧ (∀ i ∈ DOMAIN seq, j ∈ DOMAIN seq : i < j ⇒ seq[i] < seq[j])

46 DeleteElement(seq, index) ≜
47   [i ∈ 1 .. (Len(seq) − 1) ↦ IF i < index THEN seq[i] ELSE seq[(i + 1)]]
```

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

```
52 RECURSIVE Retain(_, _)
53 Retain(seq, R) ≜
54     IF seq = ⟨⟩
55      THEN ⟨⟩
56      ELSE LET h ≜ Head(seq)
57               IN  IF h ∈ R
```

```
58                  THEN ⟨h⟩ ∘ Retain(Tail(seq), R)
59                  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)

```
67  InsertElement(seq, elem, index) ≜
68    [i ∈ 1 .. (Len(seq) + 1) ↦ IF i < index
69                               THEN IF i = (Len(seq) + 1)
70                                    THEN elem
71                                    ELSE seq[i]
72                               ELSE IF i = index
73                                    THEN elem
74                                    ELSE seq[(i − 1)]]  i > index

76  IsSorted2Partition(n, seq1, seq2) ≜
77     ∧ seq1 ∈ Seq(1 .. n)
78     ∧ seq2 ∈ Seq(1 .. n)
79     ∧ n = Len(seq1) + Len(seq2)
80     ∧ (∀ i ∈ DOMAIN seq1, j ∈ DOMAIN seq1 : i < j ⇒ seq1[i] < seq1[j])
81     ∧ (∀ i ∈ DOMAIN seq2, j ∈ DOMAIN seq2 : i < j ⇒ seq2[i] < seq2[j])
82     ∧ (∀ i ∈ DOMAIN seq1, j ∈ DOMAIN seq2 : seq1[i] ≠ seq2[j])

84  IsSequenceInterleaving(seq, subSeq1, subSeq2, indSeq1, indSeq2) ≜
85     ∧ indSeq1 ∈ Seq(Nat)
86     ∧ indSeq2 ∈ Seq(Nat)
87     ∧ IsSorted2Partition(Len(seq), indSeq1, indSeq2)
88     ∧ Len(indSeq1) = Len(subSeq1)
89     ∧ Len(indSeq2) = Len(subSeq2)
90     ∧ (∀ i ∈ DOMAIN indSeq1 : seq[(indSeq1[i])] = subSeq1[i])
91     ∧ (∀ i ∈ 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/

```
98  SeqMaxLen(S, n) ≜  UNION {[1 .. m → S] : m ∈ 0 .. n}
```

Map on a sequence.

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

```
105 SeqMap(Op(_), seq) ≜ [x ∈ DOMAIN seq ↦ Op(seq[x])]

107 PermsWithin(S) ≜  {s ∈ UNION {[1 .. m → S] : m ∈ 0 .. Cardinality(S)} : Cardinality(Range(s)) = Cardina
```

All possible permutations generated based on sequence T.

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

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

$PermutationsOf(T) \triangleq \{[x \in 1 \,..\, Len(T) \mapsto T[P[x]]] : P \in PermutationKey(Len(T))\}$

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

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

ADDED by hengxin; *Aug.* 12, 2018

```
RECURSIVE FirstIndexOfElement(_, _)
FirstIndexOfElement(seq, elem) ≜
    IF Head(seq) = elem
     THEN 1
     ELSE 1 + FirstIndexOfElement(Tail(seq), elem)
```

Get the index of the first occurence of *elem* in *seq*. It returns 0 if *elem* does not occur in *seq*.

```
RECURSIVE FirstIndexOfElementSafe(_, _)
FirstIndexOfElementSafe(seq, elem) ≜
    LET RECURSIVE FirstIndexOfElementSafeHelper(_, _, _)
        FirstIndexOfElementSafeHelper(seqh, elemh, fail) ≜
            IF seqh = ⟨⟩
             THEN 0 − fail
             ELSE IF Head(seqh) = elemh
                    THEN 1
                    ELSE 1 + FirstIndexOfElementSafeHelper(Tail(seqh), elemh, fail + 1)
     IN FirstIndexOfElementSafeHelper(seq, elem, 0)
```

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

```
Compatible(seq1, seq2) ≜
    ∨ seq1 = seq2
    ∨ LET commonElements ≜ Range(seq1) ∩ Range(seq2)
      IN ∀ e1, e2 ∈ commonElements :
           ∨ e1 = e2
           ∨ FirstIndexOfElement(seq1, e1) < FirstIndexOfElement(seq1, e2)
             ≡ 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

```
RECURSIVE LCS(_, _)
LCS(seq1, seq2) ≜
     IF seq1 = ⟨⟩ ∨ seq2 = ⟨⟩
      THEN 0
      ELSE IF Last(seq1) = Last(seq2)
             THEN 1 + LCS(AllButLast(seq1), AllButLast(seq2))
```

```
173                 ELSE  MaxOfSet({LCS(AllButLast(seq1), seq2), LCS(seq1, AllButLast(seq2))})

175 LCSCompatible(seq1, seq2) ≜
176     Compatible(seq1, seq2) ≡ LCS(seq1, seq2) = Cardinality(Range(seq1) ∩ Range(seq2))

178 LCSCompatibleTest(S) ≜
179     ∀ seq1, seq2 ∈ PermsWithin(S) : LCSCompatible(seq1, seq2)
180 ╚═══════════════════════════════════════════════════════════════════════════
```

* Modification History
* *Last* modified *Tue Dec* 04 19:42:23 *CST* 2018 by *hengxin*
* Created *Tue Jul* 03 15:21:02 *CST* 2018 by *hengxin*