MODULE *Quicksort*

This module contains an abstract version of the *Quicksort* algorithm. If you are not already familiar with that algorithm, you should look it up on the Web and understand how it works–including what the partition procedure does, without worrying about how it does it. The version presented here does not specify a partition procedure, but chooses in a single step an arbitrary value that is the result that any partition procedure may produce.

The module also has a structured informal proof of *Quicksort*'s partial correctness property–namely, that if it terminates, it produces a sorted permutation of the original sequence. As described in the note "Proving Safety Properties", the proof uses the *TLAPS* proof system to check the decomposition of the proof into substeps, and to check some of the substeps whose proofs are trivial.

The version of *Quicksort* described here sorts a finite sequence of integers. It is one of the examples in Section 7.3 of "Proving Safety Properties", which is at

http://*lamport.azurewebsites.net*/tla/proving-*safety.pdf*

$$\text{EXTENDS } Integers,\ Sequences,\ FiniteSets,\ TLAPS,\ SequenceTheorems$$

This statement imports some standard modules, including ones used by the *TLAPS* proof system.

To aid in model checking the spec, we assume that the sequence to be sorted are elements of a set *Values* of integers.

$$\begin{array}{l} \text{CONSTANT } Values \\ \text{ASSUME } ValAssump \triangleq Values \subseteq Int \end{array}$$

We define $PermsOf(s)$ to be the set of permutations of a sequence s of integers. In TLA+, a sequence is a function whose domain is the set $1 \,..\, Len(s)$. A permutation of s is the composition of s with a permutation of its domain. It is defined as follows, where:

- $Automorphisms(S)$ is the set of all permutations of S, if S is a finite set–that is all functions f from S to S such that every element y of S is the image of some element of S under f.
- $f ** g$ is defined to be the composition of the functions f and g.

In TLA+, DOMAIN f is the domain of a function f.

$$\begin{array}{l} PermsOf(s) \triangleq \\ \quad \text{LET } Automorphisms(S) \triangleq \{f \in [S \to S] : \\ \qquad\qquad\qquad\qquad\qquad \forall y \in S : \exists x \in S : f[x] = y\} \\ \qquad\quad f ** g \triangleq [x \in \text{DOMAIN } g \mapsto f[g[x]]] \\ \quad \text{IN} \quad \{s ** f : f \in Automorphisms(\text{DOMAIN } s)\} \end{array}$$

We define $Max(S)$ and $Min(S)$ to be the maximum and minimum, respectively, of a finite, non-empty set S of integers.

$$\begin{array}{l} Max(S) \triangleq \text{CHOOSE } x \in S : \forall y \in S : x \geq y \\ Min(S) \triangleq \text{CHOOSE } x \in S : \forall y \in S : x \leq y \end{array}$$

The operator *Partitions* is defined so that if I is an interval that's a subset of $1 \,..\, Len(s)$ and $p \in Min(I) \,..\, Max(I) - 1$, the $Partitions(I,\ p,\ seq)$ is the set of all new values of sequence seq that a partition procedure is allowed to produce for the subinterval I using the pivot index p. That is, it's the set of all permutations of seq that leaves $seq[i]$ unchanged if i is not in I and permutes the values of $seq[i]$ for i in I so that the values for $i \leq p$ are less than or equal to the values for $i > p$.

$$
\begin{array}{l}
\mathit{Partitions}(I,\ p,\ s)\ \triangleq \\
\quad \{t \in \mathit{PermsOf}(s) : \\
\qquad \land\ \forall\, i \in (1 \mathrel{..} \mathit{Len}(s)) \setminus I : t[i] = s[i] \\
\qquad \land\ \forall\, i,\ j \in I : (i \leq p) \land (p < j) \Rightarrow (t[i] \leq t[j])\}
\end{array}
$$

Our algorithm has three variables:

seq : The array to be sorted.

*seq*0 : Holds the initial value of *seq*, for checking the result.

U : A set of intervals that are subsets of $1 \mathrel{..} \mathit{Len}(\mathit{seq}0)$, an interval being a nonempty set I of integers that equals $\mathit{Min}(I) \mathrel{..} \mathit{Max}(I)$. Initially, U equals the set containing just the single interval consisting of the entire set $1 \mathrel{..} \mathit{Len}(\mathit{seq}0)$.

The algorithm repeatedly does the following:

- Chose an arbitrary interval I in U.

- If I consists of a single element, remove I from U.

- Otherwise :
  - Let $I1$ be an initial interval of I and $I2$ be the rest of I.
  - Let *newseq* be an array that's the same as *seq* except that the elements $\mathit{seq}[x]$ with x in I are permuted so that $\mathit{newseq}[y] \leq \mathit{newseq}[z]$ for any y in $I1$ and z in $I2$.
  - Set *seq* to *newseq*.
  - Remove I from U and add $I1$ and $I2$ to U.

It stops when U is empty. Below is the algorithm written in *PlusCal*.

```
***************************************************************************
--fair algorithm Quicksortf
  variables  $seq \in Seq(Values) \setminus \{\langle\rangle\}$, $seq0 = seq$,   $U = \{1 \mathrel{..} Len(seq)\}$ ;
  f a: while ( $U \neq \{\}$ )
        f with ( $I \in U$ )
            f if ( $Cardinality(I) = 1$ )
                f $U := U \setminus \{I\}$ g
              else
                f with ( $p \in Min(I) \mathrel{..} (Max(I) - 1)$,
                         $I1 = Min(I) \mathrel{..} p$,
                         $I2 = (p + 1) \mathrel{..} Max(I)$,
                         $newseq \in Partitions(I, p, seq)$ )
                    f $seq := newseq$ ;
                      $U := (U \setminus \{I\}) \cup \{I1, I2\}$ g       g   g   g   g   g
***************************************************************************
```

Below is the TLA+ translation of the *PlusCal* code.

BEGIN TRANSLATION

VARIABLES $\mathit{seq},\ \mathit{seq}0,\ U,\ \mathit{pc}$

$\mathit{vars}\ \triangleq\ \langle \mathit{seq},\ \mathit{seq}0,\ U,\ \mathit{pc} \rangle$

```
$Init \triangleq$   Global variables
          $\land\ seq \in Seq(Values) \setminus \{\langle\rangle\}$
          $\land\ seq0 = seq$
          $\land\ U = \{1 \,..\, Len(seq)\}$
          $\land\ pc = \text{“a”}$

$a \triangleq \land\ pc = \text{“a”}$
      $\land$ IF $U \neq \{\}$
            THEN $\land\ \exists\, I \in U :$
                        IF $Cardinality(I) = 1$
                           THEN $\land\ U' = U \setminus \{I\}$
                                $\land\ seq' = seq$
                           ELSE $\land\ \exists\, p \in Min(I) \,..\, (Max(I) - 1) :$
                                      LET $I1 \triangleq Min(I) \,..\, p$IN
                                       LET $I2 \triangleq (p + 1) \,..\, Max(I)$IN
                                        $\exists\, newseq \in Partitions(I,\ p,\ seq) :$
                                          $\land\ seq' = newseq$
                                          $\land\ U' = ((U \setminus \{I\}) \cup \{I1,\ I2\})$
                 $\land\ pc' = \text{“a”}$
            ELSE $\land\ pc' = \text{“Done”}$
                 $\land$ UNCHANGED $\langle seq,\ U\rangle$
      $\land\ seq0' = seq0$

$Next \triangleq a$
            $\lor$   Disjunct to prevent deadlock on termination
              $(pc = \text{“Done”} \land$ UNCHANGED $vars)$

$Spec \triangleq \land\ Init \land \Box[Next]_{vars}$
          $\land\ \mathrm{WF}_{vars}(Next)$

$Termination \triangleq \Diamond(pc = \text{“Done”})$

 END TRANSLATION
```

PCorrect is the postcondition invariant that the algorithm should satisfy. You can use *TLC* to check this for a model in which *Seq*(*S*) is redefined to equal the set of sequences of at elements in *S* with length at most 4. A little thought shows that it then suffices to let *Values* be a set of 4 integers.

```
$PCorrect \triangleq (pc = \text{“Done”}) \Rightarrow$
                  $\land\ seq \in PermsOf(seq0)$
                  $\land\ \forall\, p,\ q \in 1 \,..\, Len(seq) : p < q \Rightarrow seq[p] \leq seq[q]$
```

Below are some definitions leading up to the definition of the inductive invariant *Inv* used to prove the postcondition *PCorrect*. The partial TLA+ proof follows. As explained in “Proving Safety Properties”, you can use *TLC* to check the level $- \langle 1 \rangle$ proof steps. *TLC* can do those checks on a model in which all sequences have length at most 3.

```
$UV \triangleq U \cup \{\{i\} : i \in 1 \,..\, Len(seq) \setminus$ UNION $U\}$
```

```
$DomainPartitions \triangleq \{DP \in \text{SUBSET SUBSET } (1 \mathrel{..} Len(seq0)) :$
                                 $\land (\text{UNION } DP) = 1 \mathrel{..} Len(seq0)$
                                 $\land \forall I \in DP : I = Min(I) \mathrel{..} Max(I)$
                                 $\land \forall I, J \in DP : (I \neq J) \Rightarrow (I \cap J = \{\})\}$

$RelSorted(I, J) \triangleq \forall i \in I, j \quad \in J : (i < j) \Rightarrow (seq[i] \leq seq[j])$

$TypeOK \triangleq \land seq \in Seq(Values) \setminus \{\langle\rangle\}$
           $\land seq0 \in Seq(Values) \setminus \{\langle\rangle\}$
           $\land U \in \text{SUBSET } ((\text{SUBSET } (1 \mathrel{..} Len(seq0))) \setminus \{\{\}\})$
           $\land pc \in \{\text{“a”}, \text{“Done”}\}$

$Inv \triangleq \land TypeOK$
        $\land (pc = \text{“Done”}) \Rightarrow (U = \{\})$
        $\land UV \in DomainPartitions$
        $\land seq \in PermsOf(seq0)$
        $\land \text{UNION } UV = 1 \mathrel{..} Len(seq0)$
        $\land \forall I, J \in UV : (I \neq J) \Rightarrow RelSorted(I, J)$

THEOREM $Spec \Rightarrow \Box PCorrect$
$\langle 1 \rangle 1.$ $Init \Rightarrow Inv$
  $\langle 2 \rangle$ SUFFICES ASSUME $Init$
                PROVE $Inv$
    OBVIOUS
  $\langle 2 \rangle 1.$ $TypeOK$
    $\langle 3 \rangle 1.$ $seq \in Seq(Values) \setminus \{\langle\rangle\}$
      BY DEF $Init, Inv, TypeOK, DomainPartitions, RelSorted, UV$
    $\langle 3 \rangle 2.$ $seq0 \in Seq(Values) \setminus \{\langle\rangle\}$
      BY DEF $Init, Inv, TypeOK, DomainPartitions, RelSorted, UV$
    $\langle 3 \rangle 3.$ $U \in \text{SUBSET } ((\text{SUBSET } (1 \mathrel{..} Len(seq0))) \setminus \{\{\}\})$
      $\langle 4 \rangle 1.$ $Len(seq0) \in Nat \land Len(seq0) > 0$
        BY $\langle 3 \rangle 1, EmptySeq, LenProperties$ DEF $Init$
      $\langle 4 \rangle 2.$ $1 \mathrel{..} Len(seq0) \neq \{\}$
        BY $\langle 4 \rangle 1$
      $\langle 4 \rangle 3.$ QED
        BY $\langle 4 \rangle 2, U = \{1 \mathrel{..} Len(seq0)\}$ DEF $Init$
    $\langle 3 \rangle 4.$ $pc \in \{\text{“a”}, \text{“Done”}\}$
      BY DEF $Init, Inv, TypeOK, DomainPartitions, RelSorted, UV$
    $\langle 3 \rangle 5.$ QED
      BY $\langle 3 \rangle 1, \langle 3 \rangle 2, \langle 3 \rangle 3, \langle 3 \rangle 4$ DEF $TypeOK$
  $\langle 2 \rangle 2.$ $pc = \text{“Done”} \Rightarrow U = \{\}$
    BY DEF $Init$
  $\langle 2 \rangle 3.$ $UV \in DomainPartitions$
    $\langle 3 \rangle 1.$ $UV = \{1 \mathrel{..} Len(seq0)\}$
      Follows easily from definition of $UV$, $seq0 = seq$, and $seq$ a non-empty sequence.
    $\langle 3\rangle 2.$ $UV \in$ SUBSET SUBSET $(1 \,..\, Len(seq0))$
      BY $\langle 3\rangle 1$ DEF $Inv$
    $\langle 3\rangle 3.$ (UNION $UV) = 1 \,..\, Len(seq0)$
      BY $\langle 3\rangle 1$
    $\langle 3\rangle 4.$ $1 \,..\, Len(seq0) = Min(1 \,..\, Len(seq0)) \,..\, Max(1 \,..\, Len(seq0))$
      Because $seq0 = seq$ and $seq$ a non-empty sequence imply $Len(seq0)$ a positive natural number.
    $\langle 3\rangle 5.$ $\forall I, J \in UV : I = J$
      BY $\langle 3\rangle 1$
    $\langle 3\rangle 6.$ QED
      BY $\langle 3\rangle 1$, $\langle 3\rangle 2$, $\langle 3\rangle 3$, $\langle 3\rangle 4$, $\langle 3\rangle 5$ DEF $DomainPartitions$
  $\langle 2\rangle 4.$ $seq \in PermsOf(seq0)$
    $\langle 3\rangle 1.$ $seq \in PermsOf(seq)$
      This is obvious because the identity function is a permutation of $1 \,..\, Len(seq)$.
    $\langle 3\rangle 2.$ QED
      BY $\langle 3\rangle 1$ DEF $Init$ , $Inv$, $TypeOK$, $DomainPartitions$, $RelSorted$, $UV$, $PermsOf$
  $\langle 2\rangle 5.$ UNION $UV = 1 \,..\, Len(seq0)$
    BY DEF $Init$, $Inv$, $TypeOK$, $DomainPartitions$, $RelSorted$, $UV$
  $\langle 2\rangle 6.$ $\forall I, J \in UV : (I \neq J) \Rightarrow RelSorted(I, J)$
    BY DEF $Init$, $Inv$, $TypeOK$, $DomainPartitions$, $RelSorted$, $UV$
  $\langle 2\rangle 7.$ QED
    BY $\langle 2\rangle 1$, $\langle 2\rangle 2$, $\langle 2\rangle 3$, $\langle 2\rangle 4$, $\langle 2\rangle 5$, $\langle 2\rangle 6$ DEF $Inv$
$\langle 1\rangle 2.$ $Inv \wedge [Next]_{\mathsf{vars}} \Rightarrow Inv'$
  $\langle 2\rangle$ SUFFICES ASSUME $Inv$,
                              $[Next]_{\mathsf{vars}}$
                       PROVE  $Inv'$
    OBVIOUS
  $\langle 2\rangle 1.$CASE $a$
    $\langle 3\rangle$ USE $\langle 2\rangle 1$
    $\langle 3\rangle 1.$CASE $U \neq \{\}$
      $\langle 4\rangle 1.$ $\wedge\ pc = $ "a"
            $\wedge\ pc' = $ "a"
        BY $\langle 3\rangle 1$ DEF $a$
      $\langle 4\rangle 2.$ PICK $I \in U : a!2!2!1!(I)$
        $a!2!2!1(I)$ is the formula following $\exists\, I \in U :$ in the definition of a.
        BY $\langle 3\rangle 1$ DEF $a$
      $\langle 4\rangle 3.$CASE $Cardinality(I) = 1$
        $\langle 5\rangle 1.$ $\wedge\ U' = U \setminus \{I\}$
              $\wedge\ seq' = seq$
              $\wedge\ seq0' = seq0$
          BY $\langle 4\rangle 2$, $\langle 4\rangle 3$ DEF $a$
        $\langle 5\rangle 2.$ QED
          $\langle 6\rangle 1.$ $UV' = UV$
```

The action removes a singleton set $\{j\}$ from U, which adds j to the set $\{\{i\} : i \in 1 \,..\, Len(seq) \setminus \text{UNION } U\}$, thereby keeping it in UV.

```
    $\langle 6\rangle 2.\ TypeOK'$
      BY $\langle 4\rangle 1, \langle 4\rangle 3, \langle 5\rangle 1$
       DEF $Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max, UV$
    $\langle 6\rangle 3.\ ((pc = \text{"Done"}) \Rightarrow (U = \{\}))'$
      BY $\langle 4\rangle 1, \langle 4\rangle 3, \langle 5\rangle 1$
       DEF $Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max, UV$
    $\langle 6\rangle 4.\ (UV \in DomainPartitions)'$
      BY $\langle 4\rangle 1, \langle 4\rangle 3, \langle 5\rangle 1, \langle 6\rangle 1$
       DEF $Inv, TypeOK, DomainPartitions$
    $\langle 6\rangle 5.\ (seq \in PermsOf(seq0))'$
      BY $\langle 4\rangle 1, \langle 4\rangle 3, \langle 5\rangle 1$
       DEF $Inv, TypeOK, PermsOf$
    $\langle 6\rangle 6.\ (\text{UNION } UV = 1 \,..\, Len(seq0))'$
      BY $\langle 5\rangle 1, \langle 6\rangle 1$ DEF $Inv$
    $\langle 6\rangle 7.\ (\forall I\_1, J \in UV : (I\_1 \neq J) \Rightarrow RelSorted(I\_1, J))'$
      BY $\langle 4\rangle 1, \langle 4\rangle 3, \langle 5\rangle 1, \langle 6\rangle 1$
       DEF $Inv, TypeOK, RelSorted$
    $\langle 6\rangle 8.$ QED
      BY $\langle 6\rangle 2, \langle 6\rangle 3, \langle 6\rangle 4, \langle 6\rangle 5, \langle 6\rangle 6, \langle 6\rangle 7$ DEF $Inv$
$\langle 4\rangle 4.$CASE $Cardinality(I) \neq 1$
  $\langle 5\rangle 1.\ seq0' = seq0$
    BY DEF $a$
  $\langle 5\rangle$ DEFINE $I1(p) \triangleq Min(I) \,..\, p$
                 $I2(p) \triangleq (p+1) \,..\, Max(I)$
  $\langle 5\rangle 2.$ PICK $p \in Min(I) \,..\, (Max(I) - 1) :$
                 $\wedge\ seq' \in Partitions(I, p, seq)$
                 $\wedge\ U' = ((U \setminus \{I\}) \cup \{I1(p), I2(p)\})$
    BY $\langle 4\rangle 2, \langle 4\rangle 4$
  $\langle 5\rangle 3.\ \wedge\ \wedge\ I1(p) \neq \{\}$
           $\wedge\ I1(p) = Min(I1(p)) \,..\, Max(I1(p))$
           $\wedge\ I1(p) \subseteq 1 \,..\, Len(seq0)$
        $\wedge\ \wedge\ I2(p) \neq \{\}$
           $\wedge\ I2(p) = Min(I2(p)) \,..\, Max(I2(p))$
           $\wedge\ I2(p) \subseteq 1 \,..\, Len(seq0)$
        $\wedge\ I1(p) \cap I2(p) = \{\}$
        $\wedge\ I1(p) \cup I2(p) = I$
        $\wedge\ \forall i \in I1(p), j \in I2(p) : (i < j) \wedge (seq[i] \leq seq[j])$
```

Since I is in U, invariant Inv implies I is a non-empty subinterval of $1 \,..\, Len(seq)$, and the $\langle 4\rangle 4$ case assumption implies $Min(I) < Max(I)$. Therefore $I1(p)$ and $I2(p)$ are nonempty subintervals of $1 \,..\, Len(seq)$. It's clear from the definitions of $I1(p)$ and $I2(p)$ that they are disjoint sets whose union is I. The final conjunct follows from the definition of $Partitions(I, p, seq)$.

```
$\langle 5\rangle 4.\ \land Len(seq) = Len(seq')$
        $\land Len(seq) = Len(seq0)$
  By $\langle 5\rangle 2$ and definition of $Partitions$.
$\langle 5\rangle 5.$ UNION $U =$ UNION $U'$
  By $\langle 5\rangle 2$ and $\langle 5\rangle 3$, since the action removes $I$ from $U$ and adds $I1(p)$ and $I2(p)$ to it.
$\langle 5\rangle 6.\ UV' = (UV \setminus \{I\}) \cup \{I1(p),\ I2(p)\}$
  BY $\langle 5\rangle 1,\ \langle 5\rangle 2,\ \langle 5\rangle 3,\ \langle 5\rangle 4,\ \langle 5\rangle 5$  DEF $UV$
  By $\langle 5\rangle 2$, $\langle 5\rangle 3$, and definition of $UV$, since $Len(seq) = Len(seq')$.
$\langle 5\rangle 7.\ TypeOK'$
  $\langle 6\rangle 1.\ (seq \in Seq(Values) \setminus \{\langle\rangle\})'$
    By $\langle 5\rangle 2$ and definitions of $Partitions$ and $PermsOf$, since $seq$ a non-empty sequence
    of $Values$ implies $PermsOf(seq)$ is one too.
  $\langle 6\rangle 2.\ (seq0 \in Seq(Values) \setminus \{\langle\rangle\})'$
    BY $\langle 5\rangle 1$  DEF $TypeOK,\ Inv$
  $\langle 6\rangle 3.\ (U \in$ SUBSET $(($SUBSET $(1\ ..\ Len(seq0))) \setminus \{\{\}\}))'$
    By $\langle 5\rangle 2$ and $\langle 5\rangle 3$.
  $\langle 6\rangle 4.\ (pc \in \{$"a", "Done"$\})'$
    BY $\langle 4\rangle 1$
  $\langle 6\rangle 5.$ QED
    BY $\langle 6\rangle 1,\ \langle 6\rangle 2,\ \langle 6\rangle 3,\ \langle 6\rangle 4$  DEF $TypeOK$
$\langle 5\rangle 8.\ ((pc =$ "Done"$) \Rightarrow (U = \{\}))'$
  BY $\langle 4\rangle 1$
$\langle 5\rangle 9.\ (UV \in DomainPartitions)'$
  $\langle 6\rangle$ HIDE  DEF $I1,\ I2$
  $\langle 6\rangle 1.\ UV' \in$ SUBSET SUBSET $(1\ ..\ Len(seq0'))$
    BY $\langle 5\rangle 6,\ \langle 5\rangle 3,\ \langle 5\rangle 4,\ \langle 5\rangle 1$   DEF $Inv$
  $\langle 6\rangle 2.$ UNION $UV' = 1\ ..\ Len(seq0')$
    BY $\langle 5\rangle 6,\ \langle 5\rangle 3,\ \langle 5\rangle 4,\ \langle 5\rangle 1$   DEF $Inv$
  $\langle 6\rangle 3.$ ASSUME NEW $J \in UV'$
        PROVE  $J = Min(J)\ ..\ Max(J)$
    $\langle 7\rangle 1.$CASE $J \in UV$
      BY $\langle 7\rangle 1$  DEF $Inv,\ DomainPartitions$
    $\langle 7\rangle 2.$CASE $J = I1(p)$
      BY $\langle 7\rangle 2,\ \langle 5\rangle 3$
    $\langle 7\rangle 3.$CASE $J = I2(p)$
      BY $\langle 7\rangle 3,\ \langle 5\rangle 3$
    $\langle 7\rangle 4.$ QED
      BY $\langle 7\rangle 1,\ \langle 7\rangle 2,\ \langle 7\rangle 3,\ \langle 5\rangle 6$
  $\langle 6\rangle 4.$ ASSUME NEW $J \in UV'$, NEW $K \in UV'$, $J \neq K$
        PROVE  $J \cap K = \{\}$
    If $J$ and $K$ are in $UV$, then this follows from $Inv$. If one of them is in $UV$ and the
    other equals $I1(p)$ or $I2(p)$, it follows because $I1(p) \cup I2(p) = I$ and $I$ is disjoint
    from other elements of $UV$. If $J$ and $K$ are $I1(p)$ and $I2(p)$, then it follows from
    the definitions of $I1(p)$ and $I2(p)$. By $\langle 5\rangle 6$, this covers all possibilities.
```

```
      $\langle 6\rangle 5.$ QED
        BY $\langle 6\rangle 1$, $\langle 6\rangle 2$, $\langle 6\rangle 3$, $\langle 6\rangle 4$  DEF $DomainPartitions$, $Min$, $Max$
    $\langle 5\rangle 10.$ $(seq \in PermsOf(seq0))'$
      By $\langle 5\rangle 2$ and definition of $Partitions$, $seq' \in PermsOf(seq)$, and $seq \in PermsOf(seq0)$
      implies $PermsOf(seq) = PermsOf(seq0)$.
    $\langle 5\rangle 11.$ $($UNION $UV = 1 \,..\, Len(seq0))'$
      $\langle 6\rangle$ HIDE  DEF $I1$, $I2$
      $\langle 6\rangle$ QED
        BY $\langle 5\rangle 6$, $\langle 5\rangle 3$, $\langle 5\rangle 4$, $\langle 5\rangle 1$   DEF $Inv$
    $\langle 5\rangle 12.$ $(\forall I\_1, J \in UV : (I\_1 \neq J) \Rightarrow RelSorted(I\_1, J))'$
      $\langle 6\rangle$ SUFFICES ASSUME NEW $I\_1 \in UV'$, NEW $J \in UV'$,
                                $(I\_1 \neq J)'$,
                                NEW $i \in I\_1'$, NEW $j \in J'$,
                                $(i < j)'$
                        PROVE   $(seq[i] \leq seq[j])'$
        BY  DEF $RelSorted$
      $\langle 6\rangle$ QED
        IF  $I\_1$ and $J$ are in $UV$, then this follows from $Inv$. If one of them is in $UV$ and
        the other equals $I1(p)$ or $I2(p)$, it follows from $Inv$ because $RelSorted(I, K)$ and
        $RelSorted(K, I)$ holds for all $K$ in $UV$ and $I1(p)$ and $I2(p)$ are subsets of I. If $I\_1$
        and $J$ are $I1(p)$ and $I2(p)$, then it follows from the definitions of $I1$ and $I2$. By
        $\langle 5\rangle 6$, this covers all possibilities.
    $\langle 5\rangle 13.$ QED
      BY $\langle 5\rangle 7$, $\langle 5\rangle 8$, $\langle 5\rangle 9$, $\langle 5\rangle 10$, $\langle 5\rangle 11$, $\langle 5\rangle 12$  DEF $Inv$
  $\langle 4\rangle 5.$ QED
    BY $\langle 4\rangle 3$, $\langle 4\rangle 4$
 $\langle 3\rangle 2.$CASE $U = \{\}$
  $\langle 4\rangle$ USE $\langle 3\rangle 2$  DEF $a$, $Inv$, $TypeOK$, $DomainPartitions$, $PermsOf$, $RelSorted$, $Min$, $Max$, $UV$
  $\langle 4\rangle 1.$ $TypeOK'$
    OBVIOUS
  $\langle 4\rangle 2.$ $((pc = \text{"Done"}) \Rightarrow (U = \{\}))'$
    OBVIOUS
  $\langle 4\rangle 3.$ $(UV \in DomainPartitions)'$
    OBVIOUS
  $\langle 4\rangle 4.$ $(seq \in PermsOf(seq0))'$
    OBVIOUS
  $\langle 4\rangle 5.$ $($UNION $UV = 1 \,..\, Len(seq0))'$
    OBVIOUS
  $\langle 4\rangle 6.$ $(\forall I, J \in UV : (I \neq J) \Rightarrow RelSorted(I, J))'$
    OBVIOUS
  $\langle 4\rangle 7.$ QED
    BY $\langle 4\rangle 1$, $\langle 4\rangle 2$, $\langle 4\rangle 3$, $\langle 4\rangle 4$, $\langle 4\rangle 5$, $\langle 4\rangle 6$  DEF $Inv$
 $\langle 3\rangle 3.$ QED
   BY $\langle 3\rangle 1$, $\langle 3\rangle 2$
$\langle 2\rangle 2.$CASE UNCHANGED $vars$
```

```
    $\langle 3\rangle 1.$ $TypeOK'$
      BY $\langle 2\rangle 2$ DEF $vars, Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max$
    $\langle 3\rangle 2.$ $((pc = \text{"Done"}) \Rightarrow (U = \{\}))'$
      BY $\langle 2\rangle 2$ DEF $vars, Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max$
    $\langle 3\rangle 3.$ $(UV \in DomainPartitions)'$
      BY $\langle 2\rangle 2$ DEF $vars, Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max, UV$
    $\langle 3\rangle 4.$ $(seq \in PermsOf(seq0))'$
      BY $\langle 2\rangle 2$ DEF $vars, Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max$
    $\langle 3\rangle 5.$ $(\text{UNION } UV = 1 \,..\, Len(seq0))'$
      BY $\langle 2\rangle 2$ DEF $vars, Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max, UV$
    $\langle 3\rangle 6.$ $(\forall I, J \in UV : (I \neq J) \Rightarrow RelSorted(I, J))'$
      BY $\langle 2\rangle 2$ DEF $vars, Inv, TypeOK, DomainPartitions, PermsOf, RelSorted, Min, Max, UV$
    $\langle 3\rangle 7.$ QED
      BY $\langle 3\rangle 1, \langle 3\rangle 2, \langle 3\rangle 3, \langle 3\rangle 4, \langle 3\rangle 5, \langle 3\rangle 6$ DEF $Inv$
  $\langle 2\rangle 3.$ QED
    BY $\langle 2\rangle 1, \langle 2\rangle 2$ DEF $Next$
$\langle 1\rangle 3.$ $Inv \Rightarrow PCorrect$
  $\langle 2\rangle$ SUFFICES ASSUME $Inv$,
                       $pc = \text{"Done"}$
                PROVE  $\land\ seq \in PermsOf(seq0)$
                       $\land\ \forall p, q \in 1 \,..\, Len(seq) : p < q \Rightarrow seq[p] \leq seq[q]$
    BY DEF $PCorrect$
  $\langle 2\rangle 1.$ $seq \in PermsOf(seq0)$
    BY DEF $Inv$
  $\langle 2\rangle 2.$ $\forall p, q \in 1 \,..\, Len(seq) : p < q \Rightarrow seq[p] \leq seq[q]$
    $\langle 3\rangle$ SUFFICES ASSUME NEW $p \in 1 \,..\, Len(seq)$, NEW $q \in 1 \,..\, Len(seq)$,
                                $p < q$
                   PROVE        $seq[p] \leq seq[q]$
      OBVIOUS
    $\langle 3\rangle 1.$ $\land\ Len(seq) = Len(seq0)$
          $\land\ Len(seq) \in Nat$
          $\land\ Len(seq) > 0$
      By $seq \in PermsOf(seq0)$, $seq$ a non-empty sequence, and definition of $PermsOf$.
    $\langle 3\rangle 2.$ $UV = \{\{i\} : i \in 1 \,..\, Len(seq)\}$
      BY $U = \{\}$ DEF $Inv, TypeOK, UV$
    $\langle 3\rangle 3.$ $\{p\} \in UV \land \{q\} \in UV$
      BY $\langle 3\rangle 1, \langle 3\rangle 2$
    $\langle 3\rangle$ QED
      BY $\langle 3\rangle 3$ DEF $Inv, RelSorted$
  $\langle 2\rangle 3.$ QED
    BY $\langle 2\rangle 1, \langle 2\rangle 2$
$\langle 1\rangle 4.$ QED
  BY $\langle 1\rangle 1, \langle 1\rangle 2, \langle 1\rangle 3, PTL$ DEF $Spec$
```

* Modification History
* Last modified *Fri* May 03 16:28:36 *PDT* 2019 by *lamport*
* Created *Mon Jun* 27 08:20:07 *PDT* 2016 by *lamport*