MODULE *SumSequence*

This module contains a trivial *PlusCal* algorithm to sum the elements of a sequence of integers, together with its non-trivial complete *TLAPS*-checked proof.

This algorithm is one of the examples in Section 7.3 of "Proving Safety Properties", which is at

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

EXTENDS *Integers*, *SequenceTheorems*, *NaturalsInduction*, *TLAPS*

To facilitate model checking, we assume that the sequence to be summed consists of integers in a set *Values* of integers.

CONSTANT *Values*

ASSUME $ValAssump \triangleq Values \subseteq Int$

In order to be able to express correctness of the algorithm, we define in TLA+ an operator *SeqSum* so that, if s is the sequence

$s\_1, \ \ldots\ , s\_n$

of integers, then $SumSeq(s)$ equals

$s\_1 + \ \ldots\ + s\_n$

The obvious TLA+ definition of *SeqSum* is

RECURSIVE $SeqSum(\_)$
$SeqSum(s) \triangleq$ IF $s = \langle\rangle$ THEN 0 ELSE $s[1] + SeqSum(Tail(s))$

However, *TLAPS* does not yet handle recursive operator definitions, but it does handle recursive function definitions. So, we define *SeqSum* in terms of a recursively defined function.

$SeqSum(s) \triangleq$
LET $SS[ss \in Seq(Int)] \triangleq$ IF $ss = \langle\rangle$ THEN 0 ELSE $ss[1] + SS[Tail(ss)]$
IN $SS[s]$

****

Here's the algorithm. It initially sets *seq* to an arbitrary sequence of integers in *Values* and leaves its value unchanged. It terminates with the variable sum equal to the sum of the elements of *seq*.

--fair algorithm *SumSequence* **{**
  variables $seq \in Seq(Values)$, $sum = 0$, $n = 1$ **;**
  { a: **while (** $n \leq Len(seq)$ **)**
      { $sum := sum + seq[n]$ **;**
        $n := n + 1$ **;** **}**
  }
}

****

BEGIN TRANSLATION

VARIABLES *seq*, *sum*, *n*, *pc*

$vars \triangleq \langle seq, sum, n, pc\rangle$

$Init \triangleq$ Global variables
  $\land\ seq \in Seq(Values)$

$\land\ sum = 0$
$\land\ n = 1$
$\land\ pc = \text{“a”}$

$a \triangleq \land\ pc = \text{“a”}$
$\land$ IF $n \leq Len(seq)$
THEN $\land\ sum' = sum + seq[n]$
$\land\ n' = n + 1$
$\land\ pc' = \text{“a”}$
ELSE $\land\ pc' = \text{“Done”}$
$\land$ UNCHANGED $\langle sum,\ n \rangle$
$\land\ seq' = seq$

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

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

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

END TRANSLATION

Correctness of the algorithm means that it satisfies these two properties:

- Safety: If it terminates, then it does so with sum equal to $SeqSum(seq)$.

- Liveness: The algorithm eventually terminates.

Safety is expressed in TLA+ by the invariance of the following postcondition.

$PCorrect \triangleq (pc = \text{“Done”}) \Rightarrow (sum = SeqSum(seq))$

To get *TLC* to check that the algorithm is correct, we use a model that overrides the definition of *Seq* so $Seq(S)$ is the set of sequences of elements of S having at most some small length. For example,

$Seq(S) \triangleq$ UNION $\{[1 \,..\, i \to S] : i \in 0 \,..\, 3\}$

is the set of such sequences with length at most 3.

The Proof of Safety

To prove the invariance of the postcondition, we need to find an inductive invariant that implies it. A suitable inductive invariant is formula *Inv* defined here.

$TypeOK \triangleq \land\ seq \in Seq(Values)$
$\land\ sum \in Int$
$\land\ n \in 1 \,..\, (Len(seq) + 1)$
$\land\ pc \in \{\text{“a”},\ \text{“Done”}\}$

$Inv \triangleq \land\ TypeOK$

$\land\ sum = SeqSum([i \in 1\,..\,(n-1) \mapsto seq[i]])$
$\land\ (pc = \text{"Done"}) \Rightarrow (n = Len(seq) + 1)$

TLC can check that *Inv* is an inductive invariant on a large enough model to give us confidence in its correctness. We can therefore try to use it to prove the postcondition.

In the course of writing the proof, *I* found that *I* needed two simple simple properties of sequences and *SeqSum*. The first essentially states that the definition of *SeqSum* is correct–that is, that it defines the operator we expect it to. TLA+ doesn't require you to prove anything when making a definition, and it allows you to write silly recursive definitions like

RECURSIVE $NotFactorial(\_)$
$NotFactorial(i) \triangleq \text{IF}\ i = 0\ \text{THEN}\ 1\ \text{ELSE}\ i * NotFactorial(i+1)$

Writing this definition doesn't mean that $NonFactorial(4)$ actually equals $4 * NonFactorial(5)$. I think it actually does, but *I*'m not sure. I do know that it doesn't imply that $NonFactorial(4)$ is a natural number. But the recursive definition of *SeqSum* is sensible, and we can prove the following lemma, which implies that $SeqSum(\langle 1, 2, 3, 4\rangle)$ equals $1 + SeqSum(\langle 2, 3, 4\rangle)$.

LEMMA $Lemma1 \triangleq$
$\forall\, s \in Seq(Int) :$
$SeqSum(s) = \text{IF}\ s = \langle\rangle\ \text{THEN}\ 0\ \text{ELSE}\ s[1] + SeqSum(Tail(s))$

What makes a formal proof of the algorithm non-trivial is that the definition of *SeqSum* essentially computes $SeqSum(seq)$ by summing the elements of *seq* from left to right, starting with $seq[1]$. However, the algorithm sums the elements from right to left, starting with $seq[Len(s)]$. Proving the correctness of the algorithm requires proving that the two ways of computing the sum produce the same result. To state that result, it's convenient to define the operator *Front* on sequences to be the mirror image of *Tail*:

$Front(\langle 1, 2, 3, 4\rangle) = \langle 2, 3, 4\rangle$

This operator is defined in the *SequenceTheorems* module. I find it more convenient to use the slightly different definition expressed by this theorem.

THEOREM $FrontDef \triangleq \forall\, S : \forall\, s \in Seq(S) :$
$Front(s) = [i \in 1\,..\,(Len(s)-1) \mapsto s[i]]$
BY DEF *Front*, *SubSeq*

LEMMA $Lemma5 \triangleq \forall\, s \in Seq(Int) :$
$(Len(s) > 0) \Rightarrow$
$(SeqSum(s) = SeqSum(Front(s)) + s[Len(s)])$

If we're interested in correctness of an algorithm, we probably don't want to spend our time proving simple properties of data types. Instead of proving these two obviously correct lemmas, it's best to check them with *TLC* to make sure we haven't made some silly mistake in writing them, and to prove correctness of the algorithm. If we want to be sure that the lemmas are correct, we can then prove them. Proofs of these lemmas are given below.

THEOREM $Spec \Rightarrow \Box PCorrect$
$\langle 1\rangle 1.\ Init \Rightarrow Inv$
$\langle 2\rangle$ SUFFICES ASSUME *Init*
PROVE *Inv*

OBVIOUS

$\langle 2 \rangle 1.$ $TypeOK$

BY $Lemma1$, $ValAssump$ DEF $Init$, Inv, $TypeOK$

$\langle 2 \rangle 2.$ $sum = SeqSum([i \in 1 \,..\, (n-1) \mapsto seq[i]])$

$\langle 3 \rangle 1.$ $(n-1) = 0$

BY DEF $Init$

$\langle 3 \rangle 2.$ $[i \in 1 \,..\, 0 \mapsto seq[i]] = \langle\rangle$

OBVIOUS

$\langle 3 \rangle 3.$ $\langle\rangle \in Seq(Int)$

OBVIOUS

$\langle 3 \rangle 4.$ QED

BY $\langle 3 \rangle 2$, $\langle 3 \rangle 1$, $\langle 3 \rangle 3$, $Lemma1$ DEF $Init$

$\langle 2 \rangle 3.$ $(pc = \text{“Done”}) \Rightarrow (n = Len(seq) + 1)$

BY $Lemma1$, $ValAssump$ DEF $Init$, Inv, $TypeOK$

$\langle 2 \rangle 4.$ QED

BY $\langle 2 \rangle 1$, $\langle 2 \rangle 2$, $\langle 2 \rangle 3$ DEF Inv

$\langle 1 \rangle 2.$ $Inv \wedge [Next]_{vars} \Rightarrow Inv'$

$\langle 2 \rangle$ SUFFICES ASSUME Inv, $[Next]_{vars}$
PROVE Inv'

OBVIOUS

$\langle 2 \rangle 1.$CASE a

$\langle 3 \rangle 1.$ $TypeOK'$

BY $\langle 2 \rangle 1$ DEF a, Inv, $TypeOK$

$\langle 3 \rangle 2.$ $(sum = SeqSum([i \in 1 \,..\, (n-1) \mapsto seq[i]]))'$

$\langle 4 \rangle 1.$CASE $n > Len(seq)$

$\langle 5 \rangle$ $\neg(n \leq Len(seq))$

BY $\langle 4 \rangle 1$ DEF Inv, $TypeOK$

$\langle 5 \rangle$ QED

BY $\langle 2 \rangle 1$, $\langle 4 \rangle 1$ DEF a, Inv, $TypeOK$

$\langle 4 \rangle 2.$CASE $n \in 1 \,..\, Len(seq)$

$\langle 5 \rangle$ DEFINE $curseq \triangleq [i \in 1 \,..\, (n-1) \mapsto seq[i]]$
$s \triangleq curseq'$

$\langle 5 \rangle$ SUFFICES $sum' = SeqSum(s)$

OBVIOUS

$\langle 5 \rangle 1.$ $\wedge\, n' - 1 = n$
$\wedge\, Len(s) = n$
$\wedge\, s[Len(s)] = seq[n]$

BY $\langle 2 \rangle 1$, $\langle 4 \rangle 2$ DEF a, Inv, $TypeOK$

$\langle 5 \rangle 2.$ $s = [i \in 1 \,..\, n \mapsto seq[i]]$

BY $\langle 5 \rangle 1$, $\langle 2 \rangle 1$ DEF a

$\langle 5 \rangle 3.$ $sum' = sum + seq[n]$

BY $\langle 2 \rangle 1$, $\langle 4 \rangle 2$ DEF a

$\langle 5 \rangle$ HIDE DEF s

$\langle 5 \rangle 4.$ $SeqSum(s) = SeqSum([i \in 1 \,..\, (Len(s)-1) \mapsto s[i]]) + s[Len(s)]$

⟨6⟩1. $\forall S,\ T : S \subseteq T \Rightarrow Seq(S) \subseteq Seq(T)$
OBVIOUS
⟨6⟩2. $seq \in Seq(Int)$
BY ⟨6⟩1, *ValAssump* DEF *Inv*, *TypeOK*
⟨6⟩3. $\forall\, i \in 1 \,..\, n : seq[i] \in Int$
BY ⟨6⟩2, ⟨4⟩2
⟨6⟩4. $s \in Seq(Int)$
BY ⟨6⟩3, ⟨5⟩2, ⟨4⟩2
⟨6⟩ QED
BY ⟨6⟩4, ⟨5⟩1, ⟨4⟩2, *Lemma5* DEF *Front*
⟨5⟩5. $curseq = [i \in 1 \,..\, (Len(s) - 1) \mapsto s[i]]$
BY ⟨5⟩1, ⟨5⟩2
⟨5⟩6. $sum = SeqSum(curseq)$
BY ⟨2⟩1, ⟨4⟩2, ⟨5⟩5 DEF *Inv*, *TypeOK*, *s*
⟨5⟩7. QED
BY ⟨5⟩1, ⟨5⟩3, ⟨5⟩4, ⟨5⟩5, ⟨5⟩6 DEF *Inv*, *TypeOK*, *s*
⟨4⟩3. QED
BY ⟨4⟩1, ⟨4⟩2 DEF *Inv*, *TypeOK*
⟨3⟩3. $((pc = \text{“Done”}) \Rightarrow (n = Len(seq) + 1))'$
BY ⟨2⟩1 DEF *a*, *Inv*, *TypeOK*
⟨3⟩4. QED
BY ⟨3⟩1, ⟨3⟩2, ⟨3⟩3 DEF *Inv*
⟨2⟩2. CASE UNCHANGED *vars*
BY ⟨2⟩2 DEF *Inv*, *TypeOK*, *vars*
⟨2⟩3. QED
BY ⟨2⟩1, ⟨2⟩2 DEF *Next*
⟨1⟩3. $Inv \Rightarrow PCorrect$
⟨2⟩ SUFFICES ASSUME Inv, $pc = \text{“Done”}$
PROVE $sum = SeqSum(seq)$
BY DEF *PCorrect*
⟨2⟩1. $seq = [i \in 1 \,..\, Len(seq) \mapsto seq[i]]$
BY DEF *Inv*, *TypeOK*
⟨2⟩2. QED
BY ⟨2⟩1 DEF *Inv*, *TypeOK*
⟨1⟩4. QED
BY ⟨1⟩1, ⟨1⟩2, ⟨1⟩3, *PTL* DEF *Spec*

Proofs of the Lemmas.

The LET definition at the heart of the definition of *SeqSum* is a standard definition of a function on sequences by tail recursion. Theorem *TailInductiveDef* of module *SequenceTheorems* proves correctness of such a definition.

LEMMA $Lemma1\_Proof \triangleq$
$\forall\, s \in Seq(Int) :$

$SeqSum(s) = \text{IF } s = \langle\rangle \text{ THEN } 0 \text{ ELSE } s[1] + SeqSum(Tail(s))$

$\langle 1\rangle$ DEFINE $DefSS(ssOfTailss,\ ss) \triangleq ss[1] + ssOfTailss$

$SS[ss \in Seq(Int)] \triangleq$

IF $ss = \langle\rangle$ THEN 0 ELSE $DefSS(SS[Tail(ss)],\ ss)$

$\langle 1\rangle 1.$ $TailInductiveDefHypothesis(SS,\ Int,\ 0,\ DefSS)$

BY DEF $TailInductiveDefHypothesis$

$\langle 1\rangle 2.$ $TailInductiveDefConclusion(SS,\ Int,\ 0,\ DefSS)$

BY $\langle 1\rangle 1,\ TailInductiveDef$

$\langle 1\rangle 3.$ $SS = [ss \in Seq(Int) \mapsto \text{IF } ss = \langle\rangle \text{ THEN } 0$
$\text{ELSE } ss[1] + SS[Tail(ss)]]$

BY $\langle 1\rangle 2$ DEF $TailInductiveDefConclusion$

$\langle 1\rangle$ QED

BY $\langle 1\rangle 3$ DEF $SeqSum$

Lemmas 2 and 3 are simple properties of *Tail* and *Front* that are used in the proof of Lemma 5.

LEMMA $Lemma2 \triangleq$

$\forall S : \forall s \in Seq(S) :$
$Len(s) > 0 \Rightarrow \land Tail(s) \in Seq(S)$
$\land Front(s) \in Seq(S)$
$\land Len(Tail(s)) = Len(s) - 1$
$\land Len(Front(s)) = Len(s) - 1$

$\langle 1\rangle$ SUFFICES ASSUME NEW S,
NEW $s \in Seq(S)$,
$Len(s) > 0$
PROVE $\land Tail(s) \in Seq(S)$
$\land Front(s) \in Seq(S)$
$\land Len(Tail(s)) = Len(s) - 1$
$\land Len(Front(s)) = Len(s) - 1$

OBVIOUS

$\langle 1\rangle 1.$ $Tail(s) \in Seq(S) \land Len(Tail(s)) = Len(s) - 1$

OBVIOUS

$\langle 1\rangle 2.$ $Front(s) \in Seq(S) \land Len(Front(s)) = Len(s) - 1$

BY $FrontDef$

$\langle 1\rangle 3.$ QED

BY $\langle 1\rangle 1,\ \langle 1\rangle 2$

LEMMA $Lemma3 \triangleq$

$\forall S : \forall s \in Seq(S) :$
$(Len(s) > 1) \Rightarrow (Tail(Front(s)) = Front(Tail(s)))$

$\langle 1\rangle$ SUFFICES ASSUME NEW S,
NEW $s \in Seq(S)$,
$Len(s) > 1$
PROVE $Tail(Front(s)) = Front(Tail(s))$

OBVIOUS

⟨1⟩1. $Tail(Front(s)) = [i \in 1 \,..\, (Len(s) - 2) \mapsto s[i + 1]]$
⟨2⟩1. $\land\ Front(s) = [i \in 1 \,..\, (Len(s) - 1) \mapsto s[i]]$
$\land\ Len(Front(s)) = Len(s) - 1$
$\land\ Front(s) \in Seq(S)$
$\land\ Len(s) \in Nat$
BY *FrontDef*
⟨2⟩2. $Len(Front(s)) > 0$
BY ⟨2⟩1
⟨2⟩3. $Front(s) \neq \langle\rangle$
BY ⟨2⟩1, ⟨2⟩2
⟨2⟩4. $Tail(Front(s)) = [i \in 1 \,..\, (Len(Front(s)) - 1) \mapsto Front(s)[i + 1]]$
⟨3⟩ DEFINE $t \triangleq Front(s)$
⟨3⟩1. $t \in Seq(S) \land t \neq \langle\rangle$
BY ⟨2⟩1, ⟨2⟩3
⟨3⟩2 $Tail(t) = [i \in 1 \,..\, (Len(t) - 1) \mapsto t[i + 1]]$
BY ⟨3⟩1
⟨3⟩3. QED
BY ⟨3⟩2
⟨2⟩5. $\forall\, i \in 0 \,..\, (Len(s) - 2) : Front(s)[i + 1] = s[i + 1]$
BY ⟨2⟩1
⟨2⟩6. $Len(Front(s)) - 1 = Len(s) - 2$
BY ⟨2⟩1
⟨2⟩7. $Tail(Front(s)) = [i \in 1 \,..\, (Len(s) - 2) \mapsto Front(s)[i + 1]]$
BY ⟨2⟩4, ⟨2⟩6
⟨2⟩8. $\forall\, i \in 1 \,..\, (Len(s) - 2) : Front(s)[i + 1] = s[i + 1]$
BY ⟨2⟩5, *Z3*
⟨2⟩9. QED
BY ⟨2⟩7, ⟨2⟩8
⟨1⟩2. $Front(Tail(s)) = [i \in 1 \,..\, (Len(s) - 2) \mapsto s[i + 1]]$
⟨2⟩1. $Tail(s) = [i \in 1 \,..\, (Len(s) - 1) \mapsto s[i + 1]]$
OBVIOUS
⟨2⟩2. QED
BY ⟨2⟩1, $Len(s) \in Nat$ DEF *Front*
⟨1⟩3. QED
BY ⟨1⟩1, ⟨1⟩2

The following lemma asserts type correctness of the *SeqSum* operator. It's proved by induction on the length of its argument. Such simple induction is expressed by theorem *NatInduction* of module *NaturalsInduction*.

LEMMA *Lemma4* $\triangleq\ \forall\, s \in Seq(Int) : SeqSum(s) \in Int$
⟨1⟩ DEFINE $P(N) \triangleq \forall\, s \in Seq(Int) : (Len(s) = N) \Rightarrow (SeqSum(s) \in Int)$
⟨1⟩1. $P(0)$
⟨2⟩ SUFFICES ASSUME NEW $s \in Seq(Int)$,
$Len(s) = 0$
PROVE $SeqSum(s) \in Int$

BY DEF P

$\langle 2 \rangle 1.\ s = \langle\rangle$

OBVIOUS

$\langle 2 \rangle$ QED

BY $\langle 2 \rangle 1$, *Lemma*1

$\langle 1 \rangle 2.$ ASSUME NEW $N \in Nat$, $P(N)$
PROVE $P(N+1)$

$\langle 2 \rangle$ SUFFICES ASSUME NEW $s \in Seq(Int)$,
$Len(s) = (N+1)$
PROVE $SeqSum(s) \in Int$

BY DEF P

$\langle 2 \rangle 1.\ s \neq \langle\rangle$

OBVIOUS

$\langle 2 \rangle 2.\ SeqSum(s) = s[1] + SeqSum(Tail(s))$

BY $\langle 2 \rangle 1$, *Lemma*1

$\langle 2 \rangle 3.\ s[1] \in Int$

BY $\langle 2 \rangle 1$

$\langle 2 \rangle 4.\ \wedge\ Len(Tail(s)) = N$
$\wedge\ Tail(s) \in Seq(Int)$

BY $\langle 2 \rangle 2$, *Lemma*2

$\langle 2 \rangle 5.\ SeqSum(Tail(s)) \in Int$

BY $\langle 1 \rangle 2$, $\langle 2 \rangle 4$

$\langle 2 \rangle 6.$ QED

BY $\langle 2 \rangle 2$, $\langle 2 \rangle 3$, $\langle 2 \rangle 5$

$\langle 1 \rangle$ HIDE DEF P

$\langle 1 \rangle 3.\ \forall\, N \in Nat : P(N)$

BY $\langle 1 \rangle 1$, $\langle 1 \rangle 2$, *NatInduction*

$\langle 1 \rangle 4.$ QED

BY $\langle 1 \rangle 3$ DEF P

LEMMA $Lemma5\_Proof \triangleq$
$\forall\, s \in Seq(Int) :$
$(Len(s) > 0) \Rightarrow$
$SeqSum(s) = SeqSum(Front(s)) + s[Len(s)]$

$\langle 1 \rangle$ DEFINE $P(N) \triangleq \forall\, s \in Seq(Int) :$
$(Len(s) = N) \Rightarrow$
$(SeqSum(s) =$ IF $Len(s) = 0$
THEN 0
ELSE $SeqSum(Front(s)) + s[Len(s)])$

$\langle 1 \rangle 1.\ P(0)$

$\langle 2 \rangle$ SUFFICES ASSUME NEW $s \in Seq(Int)$,
$Len(s) = 0$
PROVE $SeqSum(s) =$ IF $Len(s) = 0$
THEN 0

ELSE $SeqSum(Front(s)) + s[Len(s)]$

BY DEF P

$\langle 2 \rangle$ QED

BY $s = \langle\rangle$, $Lemma1$

$\langle 1 \rangle 2$. ASSUME NEW $N \in Nat$, $P(N)$

PROVE $P(N+1)$

$\langle 2 \rangle$ SUFFICES ASSUME NEW $s \in Seq(Int)$,
$Len(s) = (N+1)$
PROVE $SeqSum(s) =$ IF $Len(s) = 0$
THEN 0
ELSE $SeqSum(Front(s)) + s[Len(s)]$

BY DEF P

$\langle 2 \rangle$ SUFFICES $SeqSum(s) = SeqSum(Front(s)) + s[Len(s)]$

OBVIOUS

$\langle 2 \rangle 1$. $\land\ Front(s) \in Seq(Int)$
$\land\ Len(Front(s)) = N$

BY $Lemma2$

$\langle 2 \rangle$ DEFINE $t \triangleq Tail(s)$

$\langle 2 \rangle$ USE $FrontDef$

$\langle 2 \rangle 2$. $\land\ t \in Seq(Int)$
$\land\ Len(t) = N$
$\land\ SeqSum(s) = s[1] + SeqSum(t)$

BY $HeadTailProperties$, $Lemma1$, $s \neq \langle\rangle$

$\langle 2 \rangle 3$.CASE $N = 0$

$\langle 3 \rangle$ USE $\langle 2 \rangle 3$

$\langle 3 \rangle$ HIDE $FrontDef$ DEF $Front$

$\langle 3 \rangle 1$. $SeqSum(Front(s)) = 0$

BY $Lemma1$, $\langle 2 \rangle 1$, $Front(s) = \langle\rangle$

$\langle 3 \rangle 2$. $Len(Tail(s)) = 0$

BY $HeadTailProperties$

$\langle 3 \rangle 3$. $SeqSum(Tail(s)) =$
IF $Tail(s) = \langle\rangle$ THEN 0 ELSE $Tail(s)[1] + SeqSum(Tail(Tail(s)))$

BY $\langle 2 \rangle 2$, $Lemma1$

$\langle 3 \rangle 4$. $SeqSum(Tail(s)) = 0$

BY $\langle 3 \rangle 2$, $\langle 2 \rangle 2$, $EmptySeq$, $Tail(s) = \langle\rangle$, $\langle 3 \rangle 3$

$\langle 3 \rangle 5$. QED

BY $\langle 2 \rangle 2$, $\langle 3 \rangle 1$, $\langle 3 \rangle 4$

$\langle 2 \rangle 4$.CASE $N > 0$

$\langle 3 \rangle$ $\land\ Front(s) \in Seq(Int)$
$\land\ Front(t) \in Seq(Int)$
$\land\ Tail(Front(s)) \in Seq(Int)$

$\langle 4 \rangle 1$. $Front(s) \in Seq(Int)$

BY $\langle 2 \rangle 4$, $\langle 2 \rangle 2$, $Lemma2$

$\langle 4 \rangle 2$. $Front(t) \in Seq(Int)$

BY $\langle 2 \rangle 4$, $\langle 2 \rangle 2$, $Lemma2$

⟨4⟩3. $Tail(Front(s)) \in Seq(Int)$
⟨5⟩ $Len(s) > 1$
BY ⟨2⟩4
⟨5⟩ $Len(Front(s)) > 0$
BY $Lemma2$
⟨5⟩ $Front(s) \in Seq(Int)$
BY $Lemma2$
⟨5⟩ $Tail(Front(s)) \in Seq(Int)$
BY $Lemma2$
⟨5⟩ QED
BY $Lemma2$
⟨4⟩4. QED
BY ⟨4⟩1, ⟨4⟩2, ⟨4⟩3
⟨3⟩1. $SeqSum(t) = SeqSum(Front(t)) + \ t[N]$
BY ⟨1⟩2, ⟨2⟩2, ⟨2⟩4
⟨3⟩2. $SeqSum(t) = SeqSum(Tail(Front(s))) + t[N]$
BY ⟨3⟩1, ⟨2⟩4, $Len(s) > 1$, $Lemma3$
⟨3⟩3. $t[N] = s[N+1]$
BY ⟨2⟩2, ⟨2⟩4
⟨3⟩ HIDE DEF $Front$
⟨3⟩4. $\land\ SeqSum(s) \in Int$
$\land\ SeqSum(t) \in Int$
$\land\ SeqSum(Tail(Front(s))) \in Int$
$\land\ t[N] \in Int$
$\land\ s[1] \in Int$
⟨4⟩1. $SeqSum(s) \in Int$
BY ⟨2⟩4, ⟨2⟩2, ⟨2⟩1, $Lemma4$
⟨4⟩2. $SeqSum(t) \in Int$
BY ⟨2⟩4, ⟨2⟩2, ⟨2⟩1, $Lemma4$
⟨4⟩3. $SeqSum(Tail(Front(s))) \in Int$
⟨5⟩1. $Len(s) > 1$
BY ⟨2⟩4
⟨5⟩2. $Len(Front(s)) > 0$
BY ⟨5⟩1, $FrontDef$ DEF $Front$
⟨5⟩3. $Front(s) \neq \langle\rangle$
BY ⟨5⟩2
⟨5⟩4. $Tail(Front(s)) \in Seq(Int)$
BY ⟨5⟩3
⟨5⟩5. QED
BY ⟨2⟩4, ⟨2⟩2, ⟨2⟩1, ⟨5⟩3, $Lemma4$
⟨4⟩4. $t[N] \in Int$
BY ⟨2⟩4, ⟨2⟩2, ⟨2⟩1
⟨4⟩4a. $s[1] \in Int$
BY ⟨2⟩4
⟨4⟩5. QED

BY $\langle 4\rangle 1$, $\langle 4\rangle 2$, $\langle 4\rangle 3$, $\langle 4\rangle 4$

$\langle 3\rangle 5$. $SeqSum(s) = s[1] + SeqSum(Tail(Front(s))) + t[N]$

$\langle 4\rangle 1$. $SeqSum(s) = s[1] + SeqSum(t)$

BY $\langle 2\rangle 2$

$\langle 4\rangle 2$. QED

BY $\langle 4\rangle 1$, $\langle 3\rangle 2$, $\langle 3\rangle 4$, $Lemma4$, $Z3$

$\langle 3\rangle 6$. $t[N] = s[N+1]$

BY $\langle 2\rangle 4$

$\langle 3\rangle 7$. $s[1] = Front(s)[1]$

BY $\langle 2\rangle 4$ DEF $Front$

$\langle 3\rangle 8$. $SeqSum(Front(s)) = Front(s)[1] + SeqSum(Tail(Front(s)))$

BY $\langle 2\rangle 4$, $Lemma1$

$\langle 3\rangle 9$. QED

BY $\langle 3\rangle 5$, $\langle 3\rangle 6$, $\langle 3\rangle 7$, $\langle 3\rangle 8$

$\langle 2\rangle 5$. QED

BY $\langle 2\rangle 3$, $\langle 2\rangle 4$

$\langle 1\rangle 3$. $\forall\, N \in Nat : P(N)$

BY $\langle 1\rangle 1$, $\langle 1\rangle 2$, $NatInduction$

$\langle 1\rangle 4$. QED

BY $\langle 1\rangle 3$

* Modification History

* Last modified *Fri* May 03 16:40:42 *PDT* 2019 by *lamport*

* Created *Fri Apr* 19 14:13:06 *PDT* 2019 by *lamport*