## - MODULE MinMax1

This module and modules MinMax2 and MinMax2H are used as examples in Sections 1 and 2 of the paper "Auxiliary Variables in TLA+".

This module specifies a tiny system in which a user presents a server with a sequence of integer inputs, and the server responds to each input value i with with one of the following outputs: Hi if i is the largest number input so far, Lo if it's the smallest number input so far, Both if it's both, and None if it's neither.

The module is part of an example illustrating the use of a history variable. The example includes this module, module MinMax2, and module MinMax2H which adds a history variable to the specification of MinMax2 and shows that the resulting specification implements implements the specification of the current module under a suitable refinement mapping.

19 EXTENDS Integers

We define setMax(S) and setMin(S) to be largest and smallest value in a nonempty finite set S of integers.

```
25 setMax(S) \triangleq CHOOSE \ t \in S : \forall s \in S : t \geq s
26 setMin(S) \triangleq CHOOSE \ t \in S : \forall s \in S : t \leq s
```

The possible values that can be returned by the system are declared to be constants, which we assume are not integers.

```
32 CONSTANTS Lo, Hi, Both, None
33 ASSUME \{Lo, Hi, Both, None\} \cap Int = \{\}
```

The the value of the variable x is the value input by the user or the value output by the system, the variable turn indicating which. The variable y holds the set of all values input thus far. We consider x and turn to be externally visible and y to be internal.

```
41 VARIABLES x, turn, y
42 vars \triangleq \langle x, turn, y \rangle
```

The initial predicate *Init*:

```
\begin{array}{lll} 47 & Init \stackrel{\Delta}{=} & \wedge x = None \\ 48 & & \wedge turn = \text{``input''} \\ 49 & & \wedge y = \{\} \end{array}
```

The user's input action:

```
54 InputNum \triangleq \land turn = "input"

55 \land turn' = "output"

56 \land x' \in Int

57 \land y' = y
```

The systems response action:

```
62 Respond \triangleq \land turn = \text{``output''}

63 \land turn' = \text{``input''}

64 \land y' = y \cup \{x\}

65 \land x' = \text{If } x = setMax(y') \text{ THEN IF } x = setMin(y') \text{ THEN } Both \text{ ELSE } Hi

66 \text{ELSE IF } x = setMin(y') \text{ THEN } Lo \text{ ELSE } None
```

```
The next-state action:
```

78 F

86 F

71  $Next \triangleq InputNum \lor Respond$ 

The specification, which is a safety property (it asserts no liveness condition).

```
77 Spec \triangleq Init \wedge \Box [Next]_{vars}
```

Invariant to check (added by hengxin)

```
82 NoneCertificate \triangleq \Box[\land x \in Int]
```

83 
$$\land x < setMax(y \cup \{x\})$$
  
84  $\land x > setMin(y \cup \{x\})$ 

 $\equiv x' = None]_{vars}$ 

Below, we check that specification Spec implements specification Spec of module MinMax2 under a suitable refinement mapping. The following definitions of Infinity and MinusInfinity are copied from module MinMax2.

```
93 Infinity \triangleq \text{CHOOSE } n : n \notin Int
94 MinusInfinity \triangleq \text{CHOOSE } n : n \notin (Int \cup \{Infinity\})
```

96  $M \stackrel{\triangle}{=} \text{INSTANCE } MinMax2$ 

97 WITH 
$$min \leftarrow \text{IF } y = \{\}$$
 THEN  $Infinity$  ELSE  $setMin(y) + 1$ , 98  $max \leftarrow \text{IF } y = \{\}$  THEN  $MinusInfinity$  ELSE  $setMax(y)$ 

The following theorem asserts that Spec implements the specification Spec of module MinMax2 under the refinement mapping defined by the INSTANCE statement. The theorem can be checked with TLC using a model having M!Spec as the temporal property to be checked.

106 THEOREM  $Spec \Rightarrow M!Spec$ 

**\\*** Modification History

- \\* Last modified Wed Oct 31 19:44:20 CST 2018 by hengxin
- \\* Last modified Tue Jun 26 17:29:03 CST 2018 by hengxin
- $\backslash * Last$  modified Fri Oct 21 23:48:10 PDT 2016 by lamport
- \\* Created Fri Aug 26 14:28:26 PDT 2016 by lamport