```
─────────────────────────── MODULE GalsHeat ───────────────────────────
EXTENDS FiniteSets, Integers
───────────────────────────────────────────────────────────────────────

CONSTANTS  Nodes,           Total number of nodes
           NEIGHBOURS,      A tuple of the set of neighbours with the index
                            being the node number
           MAXTIME          Maximum time for the systems

A variable that stores the state (t, cs, ns) of each node where "t" is the current time of the node,
"cs" is the count of messages received for the current time and "ns" is the count of messages
recieved for t + 1
VARIABLE   state

───────────────────────────────────────────────────────────────────────

Initialize the state of each node so the t = 0, all the messages for t = 0 (cs) have been received
(the nodes are ready to progress to t = 1) and no messages for t = 1 (ns) have been received
GSInit  ≜  state =
              [n ∈ 1 . . Nodes ↦
                 [t ↦ 0, cs ↦ Cardinality(NEIGHBOURS[n]), ns ↦ 0]]


Next(n)  ≜  ∧ state[n].t < MAXTIME
            Check if messages for the current time have been received from all the neighbours
            ∧ state[n].cs = Cardinality(NEIGHBOURS[n])
            ∧ state′ = [i ∈ 1 . . Nodes ↦
                    IF i = n THEN
                        [  t   ↦ state[i].t + 1,
                           cs  ↦ state[i].ns,
                           ns  ↦ 0]
                     ELSE  IF i ∈ NEIGHBOURS[n] THEN
                        IF state[i].t = state[n].t + 1 THEN
                               Message received for the current time
                             [state[i] EXCEPT !.cs = 1 + @]
                          ELSE  IF state[i].t = state[n].t THEN
                               Message received for time t + 1
                             [state[i] EXCEPT !.ns = 1 + @]
                          ELSE  FALSE
                     ELSE
                        state[i]]

GSNext  ≜  ∃ n ∈ 1 . . Nodes : Next(n)

GSSpec  ≜  GSInit ∧ □[GSNext]_⟨state⟩
```

Check that the neighbours have a two way link

```
$NeighbourOK \triangleq$
        $\forall n \in 1 \,..\, Nodes :$
           $\forall a \in NEIGHBOURS[n] : n \in NEIGHBOURS[a]$
```

Check that the time difference between neighbours is never greater than 1

```
$TimeDiffOK \triangleq$
        $\forall n \in 1 \,..\, Nodes :$
           $\forall a \in NEIGHBOURS[n] :$
              $\land\ state[n].t - state[a].t < 2$
              $\land\ state[n].t - state[a].t > -2$
```

Check that the values of the state variables are correct

```
$StateVariablesOK \triangleq$
        $\forall n \in 1 \,..\, Nodes :$
           $\land\ state[n].t \leq MAXTIME$
           $\land\ state[n].cs \leq Cardinality(NEIGHBOURS[n])$
           $\land\ state[n].ns \leq Cardinality(NEIGHBOURS[n])$
```