MODULE *DieHarder*

EXTENDS *Integers*

CONSTANTS *Goal*, *Jugs*, *Capacity*

ASSUME $\land Goal \in Nat$
$\land Capacity \in [Jugs \to Nat \setminus \{0\}]$

$Min(m, n) \triangleq$ IF $m < n$ THEN m ELSE n

****

--algorithm *DieHarder*
{ variables x, $injug = [j \in Jugs \mapsto 0]$ **;**
{ while (TRUE **)**
{ either with ($j \in Jugs$ **)** fill $Jug[j]$
{ $injug[j] := Capacity[j]$ **}**
or with ($j \in Jugs$ **)** empty $Jug[j]$
{ $injug[j] := 0$ **}**

When pouring from j to k, we first ask if $In[j]+In[k] > In[k]$. If so, we make $In[k]' = Cap[k]$ and $In[j]' = In[j]-(Cap[k]-In[k])$; if not, we make $In[k]' = In[k]+In[j]$ and $In[j]' = 0$. In the former case, the amount poured is $Cap[k] - In[k]$; in the latter case, the amount poured is $In[j]$, which equals $(In[k]+In[j])-In[k]$. In both cases, the amount poured is $Min(In[k]+In[j],\ In[k])-In[k]$.

or with ($j \in Jugs,\ k \in Jugs \setminus \{j\}$ **)** pour from j to k
{ with ($poured = Min(injug[j] + injug[k],\ Capacity[k]) - injug[k]$ **)**
{ $injug[j] := injug[j] - poured$
|| $injug[k] := injug[k] + poured$
} } } } }

****

BEGIN TRANSLATION
CONSTANT *defaultInitValue*
VARIABLES x, $injug$

$vars \triangleq \langle x, injug \rangle$

$Init \triangleq$ Global variables
$\land x = defaultInitValue$
$\land injug = [j \in Jugs \mapsto 0]$

$Next \triangleq \land \lor \land \exists j \in Jugs :$
$injug' = [injug$ EXCEPT $![j] = Capacity[j]]$
$\lor \land \exists j \in Jugs :$
$injug' = [injug$ EXCEPT $![j] = 0]$
$\lor \land \exists j \in Jugs :$
$\exists k \in Jugs \setminus \{j\} :$
LET $poured \triangleq Min(injug[j] + injug[k],\ Capacity[k]) - injug[k]$ IN

$$injug' = [injug \text{ EXCEPT } ![j] = injug[j] - poured,\ ![k] = injug[k] + poured]$$

$$\land\ x' = x$$

$$Spec \triangleq Init \land \Box[Next]_{vars}$$

END TRANSLATION

* Modification History
* Last modified *Mon Feb* 17 08:13:37 *PST* 2014 by *bbeckman*
* Created *Mon Feb* 17 07:51:51 *PST* 2014 by *bbeckman*