MODULE *Euclid*

EXTENDS *Integers*, *GCD*

CONSTANTS M, N

```
****************************************************************************
--fair algorithm Euclid{
   variables x ∈ 1 .. N, y ∈ 1 .. N, x0 = x, y0 = y;
    variables x = M, y = N ;
    { judge: while ( x ≠ y ) {  do  : if ( x < y ) { y := y − x }
                                 else { x := x − y }
                               } ;
     print ⟨x, y⟩;
     assert (x = y) ∧ (x = GCD(x0, y0));
     }
}
****************************************************************************
```

BEGIN TRANSLATION ($chksum(pcal) =$ "$65d8d3f8$" $\land\ chksum(tla) =$ "$9d2756b$")

VARIABLES x, y, pc

$vars \triangleq \langle x, y, pc \rangle$

$Init \triangleq$ Global variables
$\land\ x = M$
$\land\ y = N$
$\land\ pc =$ "judge"

$judge \triangleq \land\ pc =$ "judge"
$\land$ IF $x \neq y$
THEN $\land\ pc' =$ "do"
ELSE $\land\ pc' =$ "Done"
$\land$ UNCHANGED $\langle x, y \rangle$

$do \triangleq \land\ pc =$ "do"
$\land$ IF $x < y$
THEN $\land\ y' = y - x$
$\land\ x' = x$
ELSE $\land\ x' = x - y$
$\land\ y' = y$
$\land\ pc' =$ "judge"

Allow infinite stuttering to prevent deadlock on termination.

$Terminating \triangleq pc =$ "Done" $\land$ UNCHANGED $vars$

$Next \triangleq judge \lor do$
$\lor\ Terminating$

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

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

END TRANSLATION

$Safety \triangleq (pc = \text{“Done”}) \Rightarrow (x = y) \wedge (x = GCD(M, N))$

Safety and Liveness (Termination in this part) means Correctness

* Modification History
* Last modified Sun *Sep* 26 09:36:52 *CST* 2021 by *wrz*
* Created *Fri Sep* 24 16:52:34 *CST* 2021 by *wrz*