MODULE *semaphore*

EXTENDS *Naturals*, *Sequences*, *TLC* Sequences required for "procedure" stmt
CONSTANT N N is number of iterations. Assign to it in model overview.

```
--algorithm semaphore{
  variables sem_count = 0, sem_num_waiting  = 0,
            num_sem_post = 0, num_sem_wait = 0;

  procedure sem_wait(  ) {
    wstart:     num_sem_wait := num_sem_wait + 1;  FOR DEBUGGING
    w0: sem_count := sem_count - 1;
        sem_num_waiting := sem_num_waiting + 1;
    w1: while ( TRUE ) {
    w2:   if ( sem_count > 0 ∨ sem_num_waiting > 0 - sem_count ) {
              if sem_count > 0 before, or if someone posted to us
            sem_num_waiting := sem_num_waiting - 1;
            return; } ;
    w_iswaiting: skip   ;
        } ;
   }

  procedure sem_post(  )
  { pstart:  num_sem_post := num_sem_post + 1;
    p0: sem_count := sem_count + 1;
    p1: return;
   }

  process ( thread ∈ {"thr1", "thr2"} )
    variable iterations = N;
  { start: while ( iterations > 0 ) {
      proc1: with ( choice ∈ {1, 2} )
               if ( choice  = 1 ) {
                 call sem_wait(); }
               else { call sem_post(); } ;
      proc2: iterations := iterations - 1;
      } ;  end while
     assert iterations = 0;
   }  end process block

  process ( thread3 = "cleanup" )
     Note that eventually, each thread is in "Done" or "w_iswaiting"
  { start_cleanup: while ( pc["thr1"] ≠ "Done" ∨ pc["thr2"] ≠ "Done" ) {
    clean1:          await (pc["thr1"] = "Done" ∧ pc["thr2"] = "Done") ∨
                           pc["thr1"] = "w_iswaiting" ∨ pc["thr2"] = "w_iswaiting" ;
```

```
                    if ( (pc["thr1"] = "w_iswaiting" ∨ pc["thr2"] = "w_iswaiting") ∧
                         num_sem_post < 5 ∗ N )
                      { call sem_post() ; } ;
    } ;   end while
    end_cleanup:
    assert sem_count = num_sem_post − num_sem_wait ;
     print sem_count;
    assert sem_num_waiting = 0 ;

  }   end process block

}   \ ∗ end algorithm
```