1. El programa del listado 11.7 (*PSMC, Chapter 11*) modela un planificador sin prioridades en PROMELA:

```
$ cat -n sched1_v6_a.pml | expand
     1  /* Copyright 2007 by Moti Ben-Ari under the GNU GPL; see readme.txt */
     2  /* PSMC, pp.175-177
     3     vk, 2015
     4  */
     5
     6  #define N 2             /* number of processes */
     7  byte clock = 0          /* models time */
     8  bool done[N] = false    /* done before the deadline */
     9
    10  proctype T(byte ID; byte period; byte exec) {
    11      byte next = 0       /* next time to execute */
    12      do
    13      ::  atomic {
    14              clock >= next ->    /* is it time to execute? */
    15                  printf("Task %d: executed from %d ", ID, clock)
    16                  clock = clock + exec    /* executed */
    17                  printf("to %d\n", clock)
    18                  done[ID] = true
    19                  next = next + period    /* next time to execute */
    20          }
    21      od
    22  }
    23
    24  proctype Watchdog(byte ID; byte period) {       /* for every task */
    25      byte deadline = period
    26      do
    27      ::  atomic {
    28              clock >= deadline ->
    29                  assert done[ID]
    30                  deadline = deadline + period
    31                  done[ID] = false
    32          }
    33      od
    34  }
    35
    36  proctype Idle() {
    37      do
    38      ::  atomic {
    39              timeout -> {
    40                  clock++
    41                  printf("Idle, clock ticking: %d\n", clock)
    42              }
    43          }
    44      od
    45  }
    46
    47  init {
    48      atomic {
    49          run Idle()
    50          run T(0, 2, 1)      /* Task ID, period, execution time */
    51          run Watchdog(0, 2)  /* Task ID, task deadline */
    52          run T(1, 5, 2)
    53          run Watchdog(1, 5)
    54      }
    55  }
```

Ejecutando este modelo en Spin con la semilla del generador de números aleatorios (*seed*) particular (la opción `-n`) se obtiene la siguiente salida:

```
$ spin -T -h -n1449182041 sched1_v6_a.pml
Task 0: executed from 0 to 1
Task 1: executed from 1 to 3
Task 0: executed from 3 to 4
spin: sched1_v6_a.pml:29, Error: assertion violated
spin: text of failed assertion: assert(done[ID])
#processes: 6
		clock = 4
		done[0] = 0
		done[1] = 1
 31:	proc  5 (Watchdog:1) sched1_v6_a.pml:26 (state 6)
 31:	proc  4 (T:1) sched1_v6_a.pml:22 (state 9)
 31:	proc  3 (Watchdog:1) sched1_v6_a.pml:29 (state 2)
 31:	proc  2 (T:1) sched1_v6_a.pml:22 (state 9)
 31:	proc  1 (Idle:1) sched1_v6_a.pml:37 (state 6)
 31:	proc  0 (:init::1) sched1_v6_a.pml:55 (state 7) <valid end state>
6 processes created
seed used: 1449182041
```

Pero la planificación de las tareas es correcta:

1. En el tiempo 0: la tarea T_0 (con el período de 2, en su 1er periodo) se ejecutó de 0 a 1, antes de su *deadline* que es 2.
2. En el tiempo 1: la tarea T_1 (con el período de 5, en su 1er periodo) se ejecutó de 1 a 3, antes de su *deadline* que es 5.
3. En el tiempo 3: la tarea T_0 (con el período de 2, en su 2do periodo) se ejecutó de 3 a 4, antes de su *deadline* que es 4.

Estamos en el tiempo 4, pasaron 2 períodos de la tarea T_0, y esta se ejecutó 2 veces. La tarea T_1 se ejecutó en su 1^{er} período, y todavía no comenzó su 2do período. Entonces, ¿cuál es el problema? ¿Por qué el aserto fue violado y la simulación fue abortada?

2. Analice los siguientes modelos y sus verificaciones con Spin. Presente sus conclusiones.

```
$ cat -n priority_man_example_3.pml | expand
     1  /*** Example 3 ***/
     2
     3  chan q = [1] of { bool }
     4  bool ok = false
     5
     6  active proctype high () priority 10
     7  {   bool x
     8
     9      q?x    /* highest priotity, but blocked */
    10      ok = true
    11  }
    12
    13
    14  active proctype low () priority 5
    15  {
    16      atomic {
    17          q!true      /* executes first  */
    18          assert (ok) /* assertion fails */
    19      }
    20  }
```

```
$ spin priority_man_example_3.pml
2 processes created

$ spin -i priority_man_example_3.pml
Select a statement
choice 1: proc  1 (low:5) priority_man_example_3.pml:17 (state 1) [q!1]
Select [1-2]: 1
Select a statement
choice 1: proc  0 (high:10) priority_man_example_3.pml:9 (state 1) [q?x]
Select [1-1]: 1
Select a statement
choice 1: proc  0 (high:10) priority_man_example_3.pml:10 (state 2) [ok = 1]
Select [1-1]: 1
Select a statement
choice 1: proc  1 (low:5) priority_man_example_3.pml:18 (state 2) [assert(ok)]
Select [1-2]: 1
Select a statement
choice  1:  proc   1  (low:5)  priority_man_example_3.pml:20  (state  4)  <valid  end
state> [-end-]
Select [1-2]: 1
2 processes created

$ spin -v priority_man_example_3.pml
  1:   proc  1 (low:5) priority_man_example_3.pml:17 (state 1)    [q!1]
  2:   proc  1 (low:5) priority_man_example_3.pml:18 (state 2)    <<Not Enabled>>
  2:   proc  0 (high:10) priority_man_example_3.pml:9 (state 1)   [q?x]
  3:   proc  0 (high:10) priority_man_example_3.pml:10 (state 2)  [ok = 1]
  4:   proc  1 (low:5) priority_man_example_3.pml:18 (state 2)    [assert(ok)]
2 processes created

$ spin -a priority_man_example_3.pml
$ gcc pan.c -o pan
$ ./pan | expand
hint: this search is more efficient if pan.c is compiled -DSAFETY
pan:1: assertion violated ok (at depth 0)
pan: wrote priority_man_example_3.pml.trail

(Spin Version 6.4.3 -- 16 December 2014)
Warning: Search not completed

Full statespace search for:
        never claim             - (none specified)
        assertion violations    +
        acceptance   cycles     - (not selected)
        invalid end states      +

State-vector 36 byte, depth reached 0, errors: 1
        1 states, stored
        0 states, matched
        1 transitions (= stored+matched)
        0 atomic steps
hash conflicts:         0 (resolved)

Stats on memory usage (in Megabytes):
      0.000          equivalent  memory  usage  for  states  (stored*(State-vector  +
overhead))
    0.292       actual memory usage for states
  128.000       memory used for hash table (-w24)
    0.611       memory used for DFS stack (-m10000)
  128.806       total actual memory usage

pan: elapsed time 0 seconds
```

```
$ spin -t -p -g -l priority_man_example_3.pml
using statement merging
  1:   proc  1 (low:5) priority_man_example_3.pml:17 (state 1)    [q!1]
       queue 1 (q): [1]
spin: priority_man_example_3.pml:18, Error: assertion violated
spin: text of failed assertion: assert(ok)
  1:   proc  1 (low:5) priority_man_example_3.pml:18 (state 2)    [assert(ok)]
       queue 1 (q): [1]
spin: trail ends after 1 steps
#processes: 2
       queue 1 (q): [1]
       ok = 0
  1:   proc  1 (low:5) priority_man_example_3.pml:20 (state 4) <valid end state>
  1:   proc  0 (high:10) priority_man_example_3.pml:9 (state 1)
2 processes created

$ spin -o6 priority_man_example_3.pml
spin: priority_man_example_3.pml:18, Error: assertion violated
spin: text of failed assertion: assert(ok)
#processes: 2
       queue 1 (q): [1]
       ok = 0
  2:   proc  1 (low) priority_man_example_3.pml:18 (state 2)
  2:   proc  0 (high) priority_man_example_3.pml:9 (state 1)
2 processes created

$ sed 's/atomic/d_step/g' priority_man_example_3.pml >priority_man_example_3a.pml

$ spin priority_man_example_3a.pml
spin: priority_man_example_3a.pml:18, Error: assertion violated
spin: text of failed assertion: assert(ok)
#processes: 2
       queue 1 (q): [1]
       ok = 0
  2:   proc  1 (low:5) priority_man_example_3a.pml:18 (state 2)
  2:   proc  0 (high:10) priority_man_example_3a.pml:9 (state 1)
2 processes created
```

Profesor: V. Khlebnikov
Pando, 30 de noviembre de 2018