CPDS-Conc Lab 4
Safety & Progress(II). Peterson exclusion algorithm

Authors: Jorge Castro and Joaquim Gabarro
Comments and suggestions to `gabarro@lsi.upc.edu` with header `CPDS-ConcLab` .

Goal: You have to model, check and implement in Java the Peterson algorithm as an example of a basic mutual exclusion algorithm.

Basic material:

- Course slides.
- Basic reading: Chapters 7 of the book: *Concurrency, State Models & Java Programs*, Jeff Magee and Jeff Kramer, Wiley, Second Edition, 2006 (M&K for short).

3.1 Homework

3.1.1 Peterson's exclusion algorithm

M&K 7.7, Peterson's Algorithm for two processes (Peterson, G.L., 1981). Fortunately for the warring neighbors, Gary Peterson visits one day and explains his algorithm to them. He explains that, in addition to the flags, the neighbors must share a turn indicator which can take the values 1 or 2. This is used to avoid potential progress problems.

When one neighbor wants to enter the field, he raises his flag and sets the turn indicator to indicate his neighbor turn. If he sees his neighbor's flag and it is his neighbor's turn, he can not enter but must try again later. Otherwise, he can enter the field and pick berries and must lower his flag after leaving the field.

For instance, neighbor `n1` behaves as shown below, and neighbor `n2` behaves symmetrically.

```
while (true) {
   flag1 = true; turn = 2;
   while (flag2 and turn==2) {};
   enterField; pickBerries;
   flag1 = false;
}
```

Solve the following questions:

1. Model Peterson's Algorithm for the two neighbors and provide its FSP description. Check that it does indeed satisfy the mutual exclusion (safety) and berry-picking (progress) properties, even in adverse conditions as when both neighbors are greedy.

   Hint : The following FSP can be used to model the turn indicator.

```
set CardActions = {set1,set2,[1],[2]}

CARDVAR = VAL[1],
VAL[i:Card] = ( set1 -> VAL[1]  | set2 -> VAL[2] | [i] -> VAL[i] ).
```

2. Implement in Java the warring neighbors system using the Peterson's protocol. Check that this program version avoids the progress problems present in the previous week code version when neighbors are greedy (code corresponding to exercise 7.6 of M&K).