Théorie et pratique de la concurrence – Master 1 II
TP 2 : Sections critiques (preuves)

Exercice 1 : *Variante de l'algorithme de Dekker*

Soit la variante suivante de l'algorithme de Dekker, qu'on appellera Flekker où les instructions des lignes `p6` et `q6` ont été changées :

```
boolean D1 := False // variables partagées
boolean D2 := False
int turn := 1

-- Processus P1                          -- Processus P2
loop forever :                           loop forever :
p1: section NC                           q1: section NC
p2: D1 := True                           q2: D2 := True
p3: while (D2 == True) :                 q3: while (D1 == True) :
p4:   if (turn == 2)                     q4:   if (turn == 1)
p5:     D1 := False                      q5:     D2 := False
p6:     await (D2==False) /* changée */ q6:     await (D1==False)  /* changée */
p7:     D1 := True                       q7:     D2 := True
p8: section critique                     q8: section critique
p9: turn := 2                            q9: turn := 1
p10: D1 := False                         q10: D2 := False
```

1. Modélisez l'algorithme ci-dessus en Promela et montrez avec Spin qu'il satisfait la propriété d'exclusion mutuelle.
2. Prouvez formellement que l'algorithme satisfait la propriété d'exclusion mutuelle.
3. L'algorithme de Dekker satisfait la propriété d'absence de famine et d'attente bornée sous hypothèse d'équité des processus et en supposant que toute section critique termine. Qu'en est-il de l'algorithme Flekker ? Justifiez votre réponse en utilisant Spin.

Exercice 2 : *Algorithme de Doran et Thomas*

Prouvez la correction de la variante suivante de l'algorithme de Dekker :

```
boolean D1 := False // variables partagées
boolean D2 := False
int turn := 1
```

```
-- Processus P1
loop forever :
p1:  section NC
p2:  D1 := True
p3:  while (D2 == True) :
p4:    if (turn == 2)
p5:      D1 := False
p6:      await (turn==1)
p7:      D1 := True
p8:    await (D2 == False)
p9:  section critique
p10: D1 := False
p11: turn := 2
```

```
-- Processus P2
loop forever :
q1:  section NC
q2:  D2 := True
q3:  while (D1 == True) :
q4:    if (turn == 1)
q5:      D2 := False
q6:      await (turn==2)
q7:      D2 := True
q8:    await (D1 == False)
q9:  section critique
q10: D2 := False
q11: turn := 1
```