Exercice I.1 *(Un distributeur de boissons)*
On considère un distributeur de boisson spécifié de la manière suivante :

- Une boisson coûte 1 euro.
- La machine accepte des pièces de 1 euros, 20 cents et 50 cents.
- Si la somme déjà introduite est supérieure à 1 euro, toute pièce supplémentaire est rejetée.
- Après le service la machine garde la monnaie.

Question 1. Modéliser le système sous la forme d'un automate sans variable.

Question 2. Donner un exemple d'exécution possible du modèle.

Question 3. Développer quelques branches de l'arbre des exécutions

Question 4. Proposer une autre modélisation sous la forme d'un automate à variables.

Question 5. Pour que la modélisation soit acceptable pour un outils de Model-Checking il faut que celle-ci soit finie. En affectant des propriétés à votre automate, prouver que les variables prennent bien un nombre fini de valeurs.

Exercice I.2 *(Le problème du berger)*
Pour rentrer à sa ferme, un fermier accompagné de son mouton, de son loup et d'un chou doit traverser une rivière. Pour cela, il dispose d'une minuscule barque lui permettant de transporter qu'un seul de ses compagnons à la fois (naturellement, seul le berger sait diriger la barque).

Question 1. Modéliser le système sous la forme d'un réseau synchronisé d'automates, en utilisant un automate (à 2 états) par protagoniste (berger, loup, mouton, chou). Les synchronisations autorisées représenteront les différentes traversées possibles de la barque.

Question 2. Précisez, dans votre modélisation, l'état global du système correspondant à la situation initiale, et celui correspondant à la situation que l'on souhaite atteindre.

Question 3. Par ailleurs, le berger ne peut laisser le loup seul avec le mouton, ou le mouton seul avec le chou, sinon un des deux se fait dévorer par l'autre.
Préciser les états globaux du système correspondant à ces situations interdites.

Question 4. Ecrivez une formule booléenne (formée à partir des propriétés élémentaires et des connecteurs booléens) exprimant que l'on ne se trouve pas dans un état interdit.

Question 5. Exprimer en français (ou en LTL) la propriété que le système doit vérifier correspondant au fait que le fermier puisse rentrer chez lui sans risque.

Exercice I.3 *(Le protocole de Hyman)*
Nous considérons un algorithme d'*exclusion mutuelle entre deux processus*, publié dans un journal de l'*Association for Computing Machinery* il y a une quarantaine d'années.

```
Boolean array b[0..1] initialized to true;
Integer k;
Process(i) with i either 0 or 1
begin
1 : b[i] := false;
2 : if (k!=i) then
3 :   if (not b[1-i]) then
```

```
4 :     goto 3
5 :   else
6 :     k:=i;
7 :     goto 2
8 : else
9 :   "Critical Section for i"
10:   b[i] := true;
11:   goto 1;
end
```

Dans ce programme, sont considérés comme atomiques (c'est-à-dire effectuées en une seule étape) les affectations, les évaluations de conditions et les `goto`.

Question 1. Dessinez l'automate qui correspond au processus i.

Question 2. Montrez que le programme présente un *livelock*, c'est-à-dire une exécution dans laquelle les deux processus restent indéfiniment en attente sans qu'aucun ne rentre en section critique.

Exercice I.4 *(Atomicité et famine)*
Nous poursuivons l'exercice précédent en supposant que le *booléen* k est initialisé à 0.

Question 1. Montrez que l'algorithme n'assure pas l'*exclusion mutuelle*.

Question 2. Quelles instructions faudrait-il rendre *atomiques* pour que l'exclusion mutuelle soit assurée ?

Question 3. Simplifiez l'automate, en fusionnant les transitions de `goto` avec les transitions qui les précédent, et après avoir rendu atomique les instructions de la question précédente. Vous devez obtenir un automate à 4 états, que vous nommerez I (Idle), R (Request), W (Wait) et CS (Critical Section) respectivement.

Question 4. Dessinez l'*automate global* du système de deux processus et montrez que l'exclusion mutuelle est maintenant bien assurée.

Question 5. Montrez que cet algorithme n'assure pas l'absence de *famine*, c'est-à-dire qu'il existe une exécution pour laquelle l'un des processus n'entre jamais en section critique, alors qu'il en a formulé la demande.