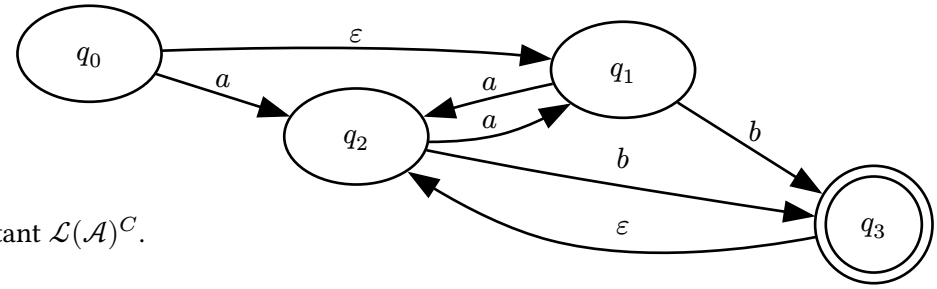
Sujet IMT-2

I - Automates

On considère l'automate $\mathcal{A}$ ci-contre.



1. Déterminiser et compléter l'automate.
2. Donner une expression régulière dénotant $\mathcal{L}(\mathcal{A})^C$.

II - Verrous à n fils d'exécution

II.a - Une première version

L'objectif est de créer un verrou pour deux fils d'execution.

On considère le pseudo-code suivant avec (`get_thread_id()` une fonction qui renvoie l'identifiant du fil courant).

```
1: function CreateLock()
2:     return {turn = 0, busy = false}

1: function Unlock(l)
2:     l.busy ← false
```

```
1: function Lock(m, t)
2:     me ← ()
3:     while l.busy do
4:         l.turn ← me
5:     l.busy ← true
```

1. Ce protocole garantit-il l'exclusion mutuelle ?

On propose une autre version de la fonction Lock :

```
1: function Lock(m, t)
2:     me ← ()
3:     while l.turn ≠ me do
4:         while l.busy do
5:             l.turn ← me
6:         l.busy ← true
```

2. Ce protocole garantit-il l'absence d'interblocage ?
3. Que dire de l'exclusion mutuelle ?

II.b - Boulangerie de Lamport

4. Expliquer précisemment le fonctionnement de l'algorithme de la boulangerie de Lamport.

5. Compléter le pseudo-code ci-dessous.

```
1: function CreateLock()
2:     want ← false,...,false
3:     ticket ← 0,...,0
4:     return {want,ticket}

1: function Unlock(m, i)
2:     m.want[i] ← false
```

```
1: function Lock(m, i)
2:     m.want[i] ← true
3:     m.ticket[i] ← max(ticket)+1
4:     while ∃j, m.want[j] = true ∧
       (m.ticket[j], j) <_lex (m.ticket[i], i) do
5:         nothing
```

6. Montrer que l'algorithme de la boulangerie de Lamport satisfait l'absence d'interblocage.
7. ... respecte l'ordre "FIFO". Qu'en déduire sur l'absence de famine ?
8. ... satisfait l'exclusion mutuelle.