

## Examen de Théorie des Langages

### Modalités

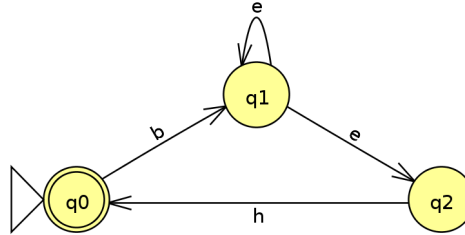
- Durée : **1h 30 minutes**
- **Il y a 3 sujets de difficultés équivalentes (sujets 0,1, et 2). Vous devez prendre le sujet dont le numéro correspond à la position dans l'alphabet de la première lettre de votre nom de famille modulo 3. ( $a \rightarrow$  sujet 0,  $b \rightarrow$  sujet 1,  $c \rightarrow$  sujet 2,  $d \rightarrow$  sujet 0, ...)**
- Aucune question au professeur n'est autorisé. Si vous pensez avoir détecté une erreur d'énoncé, expliquez les hypothèses que vous êtes amené à prendre pour continuer.
- Aucun échange, de quelque nature que ce soit, n'est autorisé.
- Le barème est donné à titre indicatif.
- Le rendu est à déposer sur Arel, dans l'onglet travaux du module, avant 15h40.
- Concernant le rendu, vous pouvez utiliser JFLAP. Dans ce cas merci de rendre une archive compressée structurée comme suit :

```
id_arel/  
  redaction.txt / .doc / .xlsx / ou autre  
  ex1.jff  
  ex2.jff  
  ex3.jff
```

- **La clarté et la précision de la rédaction seront prises en compte dans l'évaluation.**

### Exercice 1. Mouton régulier (7 points)

Soit l'automate suivant sur l'alphabet  $\{b, e, h\}$  :



**Question 1.** Cet automate est-il déterministe ? Pourquoi ? En quoi est-ce un problème ?

**Question 2.** Donner la mise en équation de l'automate et résoudre ce système.

**Question 3.** Effectuer la méthode des quotients gauches sur le langage  $L_1$  obtenu et conclure.

### Exercice 2. Mouton algébrique (4 points)

Soit le langage  $L_1$  trouvé à l'exercice précédent. Nous souhaitons maintenant trouver un sous langage  $L_2$  de celui-ci, en rajoutant la contrainte suivante : le dernier **beeeh** contient autant de **e** que les précédents réunis. Par exemple **beehbehbeehbeeeh** appartient au langage mais pas **behbeeehbeh**.

**Question 1.** Construire au choix un automate à pile ou une grammaire qui reconnaît les mots du langage  $L_2$ .

### Exercice 3. Mouton essouffé (5 points)

Nous souhaitons transformer un mot du langage  $L_1$  en ne gardant que des **beeeh** de plus en plus courts. On arrêtera la réécriture dès que la suite décroissante sera interrompue. Par exemple **beeehbeehbehbeehbeh** sera transformé en **beeehbeehbeh**.

**Question 1.** Construire une machine de Turing qui applique cette transformation aux mots du langage  $L_1$ .

### Exercice 4. Algorithme CKY (4 points)

Soit la grammaire suivante :

S  $\rightarrow$  CA | AB  
A  $\rightarrow$  b | BB  
B  $\rightarrow$  CA | a  
C  $\rightarrow$  a | AC

**Question 1.** Appliquer l'algorithme CKY pour vérifier l'appartenance du mot *baaba* à ce langage.