# Apprentissage de langages réguliers

Épreuve pratique d'algorithmique et de programmation
Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juin-juillet 2006

**ATTENTION !**

N'oubliez en aucun cas de recopier votre $u_0$
à l'emplacement prévu sur votre fiche réponse

**Important.**
Lorsque la description d'un algorithme est demandée, vous devez présenter son fonctionnement de façon schématique, courte et précise. Vous ne devez en aucun cas recopier le code de vos procédures!
Quand on demande la complexité en temps ou en mémoire d'un algorithme en fonction d'un paramètre $n$, on demande l'ordre de grandeur en fonction du paramètre, par exemple: $O(n^2)$, $O(n \log n)$,...
Il est recommandé de commencer par lancer vos programmes sur de petites valeurs des paramètres et de ***tester vos programmes sur des petits exemples que vous aurez résolus préalablement à la main.***

Ce problème est consacré à la construction d'un automate fini déterministe acceptant un ensemble fini de mots. On parle d'apprentissage lorsque le langage reconnu par l'automate généralise cet ensemble de mots (appelés « exemple ») sous la forme d'un langage régulier plus « simple ». Nous commencerons par construire un automate reconnaissant exactement l'ensemble des exemples (partie 3), puis nous étudierons une méthode de généralisation (partie 4).
A partir seulement d'exemples, il est toujours possible de généraliser au maximum, jusqu'au langage universel (reconnaissant tous les mots). Nous verrons deux méthodes pour contourner ce problème : la première consiste à généraliser jusqu'à un type de langage particulier appelé langage réversible (partie 5), la seconde, à empêcher une sur-généralisation par l'ajout de « contre-exemples », que l'automate ne doit pas accepter (partie 6).

# 1 Préliminaire

Soit la suite $(u_i)_{i\geq 0}$ définie de la façon suivante :
– $u_0$ est donné sur votre table *(à reporter sur votre fiche).*
– $u_{i+1} = (1 + u_i * 16423)$ modulo 62951.

**Question 1** *Quelle est la valeur de* ***a)*** $u_{10}$, ***b)*** $u_{2000}$, ***c)*** $u_{20000}$ *?*

# 2 Ensemble de mots

Soit $n$ un entier strictement positif. Notre alphabet $A_n$ sera l'ensemble $\{0, 1, \ldots, n-1\}$. L'ensemble des mots $A_n^*$ est donc l'ensemble des suites finies sur $A_n$.
Soit $(v_i)$ défini comme :

$$v_i = \left\lfloor \frac{u_i}{n} \right\rfloor \text{ modulo } (n+1),$$

ou $\lfloor x \rfloor$ désigne la partie entière de $x$.
Avec $k \geq 0$, nous considérons la liste des mots obtenu en découpant la séquence $(v_{2i})_{0\leq i\leq k}$ au niveau des $n$. Un mot est ainsi une séquence $v_{2a}v_{2a+2}\ldots v_{2b}$ vérifiant les trois conditions suivantes :

1. $\forall a \leq i \leq b, v_{2i} < n$,
2. $a = 0$ ou $v_{2a-2} = n$,
3. $b = k$ ou $v_{2b+2} = n$.

Lorsque $v_0 = n$ ou $v_{2k} = n$, ou si deux $n$ apparaissent consécutivement dans la séquence, cela doit apparaître dans la liste comme un mot vide.
Ainsi, par exemple, si la séquence des $v_{2i}$ pour $n = 3$ était :

$$2\ 1\ 3\ 1\ 0\ 3\ 3\ 2\ 1\ 0\ 3,$$

les mots successivement générés seraient 21, 10, $\varepsilon$ (mot vide), 210 et $\varepsilon$ (car la séquence se termine par un $n$).
Nous notons $L_{n,k}$ la liste de mots ainsi formée, dans l'ordre de leur apparition dans la séquence des $(v_i)$, et $E_{n,k}$ l'ensemble des mots distincts présents dans $L_{n,k}$.

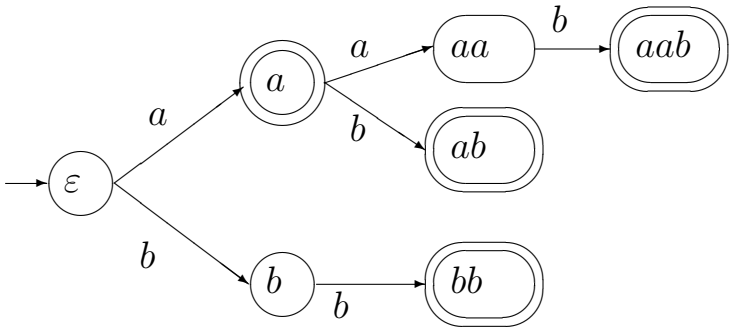



FIG. 1 – L'arbre accepteur de préfixe pour le langage $\{a, aab, ab, bb\}$. Le mot vide $\varepsilon$ est l'unique état initial. Les états finaux sont représentés par un double trait.

**Question 2** *Combien $L_{n,k}$ a-t-il d'élements avec* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$ *? Quelle est la longueur maximale des mots de $E_{n,k}$ pour* ***d)*** $n = 3$ *et* $k = 20$, ***e)*** $n = 15$ *et* $k = 600$, ***f)*** $n = 40$ *et* $k = 6000$ *?*

On rappelle qu'un automate fini peut se représenter comme un quintuplet $< Q, \Sigma, \delta, Q_0, F >$, où $Q$ est l'ensemble fini des états, $\Sigma$ l'alphabet fini, $\delta \in Q \times \Sigma \to \mathcal{P}(Q)$ (où $\mathcal{P}(Q)$ est l'ensemble des parties de $Q$) est la fonction de transition, $Q_0$ l'ensemble de états initiaux et $F$ l'ensemble des états finaux.
Nous considérerons ici essentiellement des automates déterministes : $Q_0$ est un singleton $\{q_0\}$ et pour tout état $q$ et lettre $a$, $\delta(q, a)$ ne contient au plus un seul élément.

# 3 Arbre accepteur de préfixes

L'arbre accepteur de préfixes d'un ensemble fini $E$ de mots d'un alphabet $A$ , noté PTA($E$), est un automate fini déterministe défini ainsi :
- son ensemble d'états $Q$ est l'ensemble des préfixes des mots de $E$, formellement $Q = \bigcup_{u \in E} \mathit{prefix}(u)$ ;
- ses états finaux sont les mots de E, $F = E$ ;
- son unique état intial est le mot vide, $Q_0 = \{\varepsilon\}$ ;
- pour tout $u \in Q$ et tout $a \in A$, $\delta(u, a) = \{u.a\}$ si $u.a \in Q$, et $\delta(u, a) = \emptyset$ sinon.

La figure (1) donne un exemple d'arbre accepteur de préfixes pour un langage simple.
L'arbre accepteur de préfixes d'un ensemble correspondant à une liste de mots peut être construit de façon incrémentale, en ajoutant pour chaque nouveau mot les éventuels états nécessaires pour ce mot.

**Question 3** *Combien d'états comprend* PTA($E_{n,k}$) *pour* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$ *? Quelle est la longueur du plus grand préfixe commun à deux mots distincts de $E_{n,k}$ pour* ***d)*** $n = 3$ *et* $k = 20$, ***e)*** $n = 15$ *et* $k = 600$, *et* ***f)*** $n = 40$ *et* $k = 6000$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé ainsi que sa complexité.*

Nous pourrons avoir besoin dans la suite du problème de définir un ordre sur les états de l'arbre. Notre ordre $\preceq$ sur l'ensemble $Q$ des états de PTA($E_{n,k}$) est défini ainsi : pour tout $q$,$q'$ dans $Q$, $q \preceq q'$ si :

1. $q = q'$,
2. ou $q$ est strictement moins long que $q'$,
3. ou $q$ et $q'$ sont des même longueur, et $q$ est plus petit que $q'$ lexicographiquement (au premier rang $i$ où $q(i)$ et $q'(i)$ diffèrent, $q(i) < q'(i)$).

Pour obtenir les éléments de $Q$ dans cet ordre, on peut parcourir l'arbre accepteur de préfixes en largeur, en prenant successivement la racine (l'état initial), puis ses descendants directs, puis les descendants de ses descendants, etc.

**Question 4** *Si $q_0 \prec q_1 \prec \ldots \prec q_{l-1}$ est la liste des états de* PTA($E_{n,k}$) *ordonnés de façon strictement croissante ($l$ étant le nombre de ces états), quelle est la dernière « lettre » de $q_{\lfloor \frac{l}{2} \rfloor}$ pour* ***a)*** *$n = 3$ et $k = 20$,* ***b)*** *$n = 15$ et $k = 600$, et* ***c)*** *$n = 40$ et $k = 6000$ ?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé ainsi que sa complexité.*

L'arbre accepteur de préfixes reconnaît exactement les mots de l'ensemble d'exemples. L'apprentissage ne comprend donc aucune généralisation.

# 4 Fusion d'états, fusion pour déterminisation

La méthode utilisée pour généraliser le langage sera de faire des fusions d'états à partir de l'arbre accepteur de préfixes. Étant donné un automate $\mathcal{A} =< Q, A, \delta, Q_0, F >$ et deux états $q$ et $q'$ de $Q$, l'automate généré par la fusion dans $\mathcal{A}$ de $q_1$ et $q_2$ est défini comme $\mathcal{A}' =< Q', A, \delta', Q'_0, F' >$ avec :

$$
\begin{aligned}
Q' &= (Q - \{q_1, q_2\}) \cup \{q'\} \text{ où } q' \text{ est un nouvel état} \\
Q'_0 &= \begin{cases} Q_0 & \text{si } Q_0 \cap \{q_1, q_2\} = \emptyset \\ (Q_0 - \{q_1, q_2\}) \cup \{q'\} & \text{sinon} \end{cases} \\
F' &= \begin{cases} F & \text{si } F \cap \{q_1, q_2\} = \emptyset \\ (F - \{q_1, q_2\}) \cup \{q'\} & \text{sinon} \end{cases} \\
\forall q \in Q - \{q_1, q_2\},\ \delta'(q, a) &= \begin{cases} \delta(q, a) & \text{si } \delta(q, a) \cap \{q_1, q_2\} = \emptyset \\ (\delta(q, a) - \{q_1, q_2\}) \cup \{q'\} & \text{sinon} \end{cases} \\
\delta'(q', a) &= \begin{cases} \delta(q_1, a) \cup \delta(q_2, a) \\ \qquad \text{si } (\delta(q_1, a) \cup \delta(q_2, a)) \cap \{q_1, q_2\} = \emptyset \\ (\delta(q_1, a) \cup \delta(q_2, a)) - \{q_1, q_2\} \cup \{q'\} \text{ sinon} \end{cases}
\end{aligned}
$$

La fusion de $q_1$ et $q_2$ consiste donc à remplacer les deux états par un seul état $q'$, qui cumule les propriétés (état final ou initial) et les transitions entrantes et sortantes des deux états. La figure (2) montre le résultat de la fusion de deux états à partir de l'automate de la figure (1). Noter que le langage reconnu par $\mathcal{A}'$ contient le langage reconnu par $\mathcal{A}$, et que la fusion de deux états d'un automate déterministe ne donne pas forcément un automate déterministe.

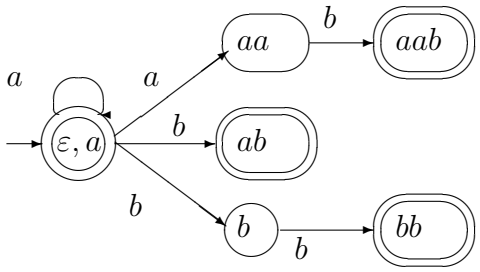



FIG. 2 – L'automate obtenu à partir de l'automate de la figure (1) par fusion de l'état $\varepsilon$ et l'état $a$. Le langage obtenu est un surensemble du langage initial, et l'automate n'est pas déterministe.

**Question 5** *Donner le nombre de couples d'états de* $\mathrm{PTA}(E_{n,k})$ *tels que la fusion de ces états donnerait un automate déterministe pour* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$ *?*

Si l'automate n'est pas déterministe, on se propose de le « déterminiser » en continuant à fusioner des états (ce procédé n'est pas la déterminisation « classique » d'un automate : le langage reconnu par l'automate n'est pas conservé par cette opération). Plus précisément, on fusionne deux états $q_1$ et $q_2$ dès que ce sont des états initiaux, ou des éléments d'un même $\delta(q, a)$.
L'algorithme s'arrête lorsque l'automate est déterministe (la fusion réduisant le nombre d'états de l'automate, cet arrêt est garanti), et l'automate obtenu est indépendant de l'ordre des fusions successives.

**Question 6** *Donner le nombre d'états de l'automate déduit de* $\mathrm{PTA}(E_{n,k})$ *par la fusion des deux plus petits états (de* $\mathrm{PTA}(E_{n,k})$*) pour l'ordre* $\preceq$*, suivi des fusions pour déterminisation, pour* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$.

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé ainsi que sa complexité.*

En fusionnant des états, on augmente le langage reconnu par l'automate, ce qui permet de généraliser l'apprentissage. A l'extrême, fusionner tous les états donne l'automate universel (qui reconnaît tous les mots possibles), ce qui constitue une surgénéralisation. On cherchera donc des critères pour limiter la fusion à certains états.

# 5 Apprentissage d'un automate bidéterministe

Étant donné un automate $\mathcal{A} =< Q, A, \delta, Q_0, F >$, l'automate miroir (ou inverse) de $\mathcal{A}$, noté $\mathcal{A}^r$, est défini par $< Q, A, \delta^r, F, Q_0 >$ avec $\delta^r(q, a) = \{q' \mid q \in \delta(q', a)\}$. Un automate est dit bidéterministe (ou 0-reversible) si il est déterministe et son miroir est déterministe. En particulier, un automate bideterministe n'a qu'un état initial et un état final.
En général, l'arbre accepteur de préfixes d'un ensemble de mots n'est pas bidéterministe (car il possède plusieurs états finaux). Néanmoins, on peut créer un automate bidéterministe à

partir de cet arbre en fusionnant les états qui empêchent ce bidéterminisme. On cherchera donc à fusionner les états $q_1$ et $q_2$ tels que :
– $q_1$ et $q_2$ sont des états finaux ;
– ou ils sont successeurs d'un même état par une même lettre ;
– ou ils sont prédecesseurs d'un même état par une même lettre.
A partir de PTA($E_{n,k}$), on peut construire ainsi un automate bidéterministe reconnaissant les éléments de $E_{n,k}$. L'automate obtenu est indépendant de l'ordre dans lequel les fusions sont faites, et est le plus grand automate bidéterministe dérivé par fusion de PTA($E_{n,k}$). On notera cet automate, BD($E_{n,k}$).

**Question 7** *Donner le nombre d'états de* BD($E_{n,k}$) *pour* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$. *Si on associe à chaque état de* BD($E_{n,k}$) *l'ensemble des états dont il est issu dans* PTA($E_{n,k}$), *de combien d'états maximum un état de* BD($E_{n,k}$) *est issu, pour* ***d)*** $n = 3$ *et* $k = 20$, ***e)*** $n = 15$ *et* $k = 600$, *et* ***f)*** $n = 40$ *et* $k = 6000$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé ainsi que sa complexité.*

La construction de BD($E_{n,k}$) permet d'obtenir le plus petit langage reconnu par un automate bidéterministe incluant $E_{n,k}$. Il s'agit donc d'un apprentissage de langages particuliers à partir d'ensembles d'exemples uniquement.

# 6 Apprentissage sous contrôle d'un échantillon négatif

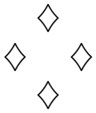
Nous cherchons maintenant un langage régulier incluant $E_{n,k}$ et disjoint avec un ensemble de contre-exemples $E^-_{n,k}$.
Nous définissons $E^-_{n,k}$ d'une façon similaire à $E_{n,k}$ : on découpe la séquence $(v_{10000+i})_{0\leq i\leq k}$ autour des $n$. $E^-_{n,k}$ est alors l'ensemble des sous-séquences qui ne sont pas déjà dans $E_{n,k}$.

**Question 8** *Donner le cardinal de* $E^-_{n,k}$ *pour pour* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$.

Pour construire un automate minimal reconnaissant tous les mots de $E_{n,k}$ et aucun de $E^-_{n,k}$, on procède par essais successifs de fusion (suivi de déterminisation par fusions). A chaque essai :
– si le nouvel automate déterministe n'accepte aucun mot $E^-_{n,k}$, il est conservé et les autres essais se font depuis cet automate ;
– sinon, on réessaie depuis l'automate précédant la fusion.
Contrairement à la construction de l'automate bi-déterministe, l'ordre des fusions est important. Nous utiliserons l'ordre $\preceq$ défini auparavant : si $q_0 \prec q_1 \prec \dots$ est la liste des éléments de $Q$ triés par ordre croissant selon $\prec$, on cherchera à fusionner $q_{i_1}$ et $q_{i_2}$, avec $i_2 > i_1$, de telle façon que $i_2$ soit le plus petit possible (et, en cas de plusieurs choix pour un même $i_2$, que $i_1$ soit le plus petit possible). Lorsqu'une fusion réussit, le nouvel état $q'$ prend dans la liste la place du plus petit état dont il est issu, et on cherche de nouveau deux états à fusionner. Notre algorithme s'arrête lorsque plus aucune fusion n'est possible sans reconnaître un mot de $E^-_{n,k}$.

**Question 9** *Donner le nombre d'opérations globales fusion + déterminisation réussies en appliquant cette méthode sur* $\mathrm{PTA}(E_{n,k})$ *pour* ***a)*** $n = 3$ *et* $k = 20$, ***b)*** $n = 15$ *et* $k = 600$, *et* ***c)*** $n = 40$ *et* $k = 6000$. *Quel est le nombre d'états de l'automate final pour* ***d)*** $n = 3$ *et* $k = 20$, ***e)*** $n = 15$ *et* $k = 600$, *et* ***f)*** $n = 40$ *et* $k = 6000$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé ainsi que sa complexité.*
*Indication : on peut constater lorsque une fusion entre deux états a échoué, toute tentative sur des états issus ultérieurement de ces deux états échouera également.*



# Apprentissage de langages réguliers

Nom, prénom, $u_0$: .....

**Question 1**

a) ☐

b) ☐

c) ☐

**Question 2**

a) ☐

b) ☐

c) ☐

d) ☐

e) ☐

f) ☐

**Question 3**

a) ☐

b) ☐

c) ☐

d) ☐

e) ☐

f) ☐

**Question 4**

a) ☐

b) ☐

c) ☐

**Question 5**

a) ☐

b) ☐

c) ☐

**Question 6**

a) ☐

b) ☐

c) ☐

**Question 7**

a) ☐

b) ☐

c) ☐

d) ☐

e) ☐

f) ☐

**Question 8**

a) ☐

b) ☐

c) ☐

**Question 9**

a) ☐

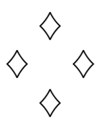
b) ☐

c) ☐

d) ☐

e) ☐

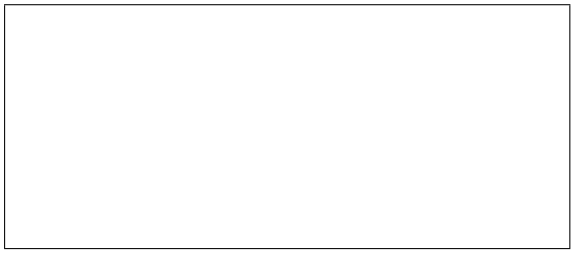
f) ☐



# Fiche d'évaluation: Apprentissage de langages réguliers

Nom, prénom, $u_0$: .....

**Question 1** 0–1–2–3–4–5–6–7–8

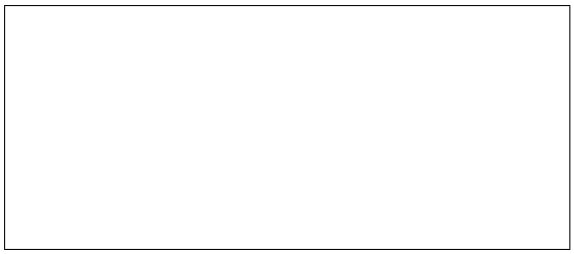


a) 0 ⊢——+——+——+——⊣ 4

b) 0 ⊢——+——+——+——⊣ 4

c) 0 ⊢——+——+——+——⊣ 4

**Question 2** 0–1–2–3–4–5–6–7–8



a) 0 ⊢——+——+——+——⊣ 4

b) 0 ⊢——+——+——+——⊣ 4

c) 0 ⊢——+——+——+——⊣ 4

d) 0 ⊢——+——+——+——⊣ 4

e) 0 ⊢——+——+——+——⊣ 4

f) 0 ⊢——+——+——+——⊣ 4

**Question 3** 0–1–2–3–4–5–6–7–8

a) 0 ⊢——+——+——+——⊣ 4

b) 0 ⊢——+——+——+——⊣ 4

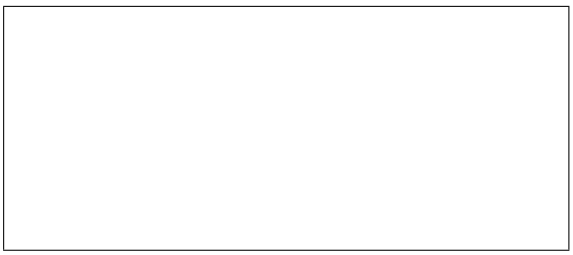
c) 0 ⊢——+——+——+——⊣ 4

d) 0 ⊢——+——+——+——⊣ 4

e) 0 ⊢——+——+——+——⊣ 4

f) 0 ⊢——+——+——+——⊣ 4

**Question 4** 0–1–2–3–4–5–6–7–8

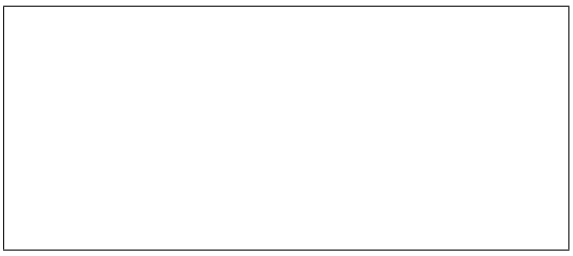


a) 0 ⊢——+——+——+——⊣ 4

b) 0 ⊢——+——+——+——⊣ 4

c) 0 ⊢——+——+——+——⊣ 4

**Question 5** 0–1–2–3–4–5–6–7–8



a) 0 ⊢——+——+——+——⊣ 4

b) 0 ⊢——+——+——+——⊣ 4

c) 0 ⊢——+——+——+——⊣ 4

**Question 6** 0–1–2–3–4–5–6–7–8

a) 0 ⊢——+——+——+——⊣ 4

b) 0 4

c) 0 4

**Question 7** 0 – 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8

a) 0 4

b) 0 4

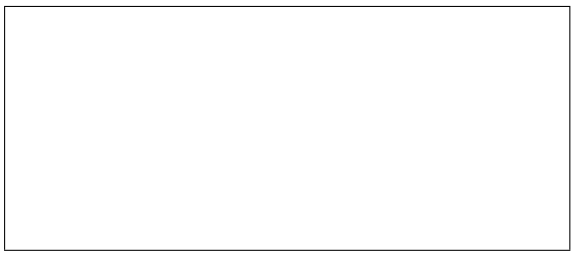
c) 0 4

d) 0 4

e) 0 4

f) 0 4

**Question 8** 0 – 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8

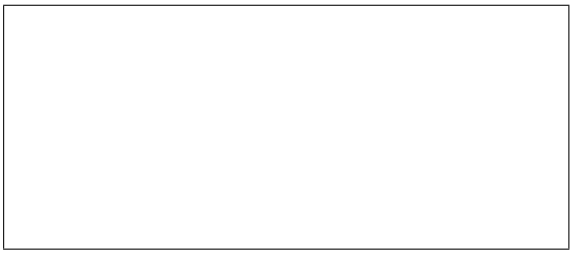


a) 0 4

b) 0 4

c) 0 4

**Question 9** 0 – 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8



a) 0 4

b) 0 4

c) 0 4

d) 0 4

e) 0 4

f) 0 4