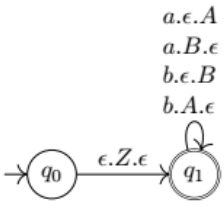
Commencé le	vendredi 5 mai 2023, 14:58
État	Terminé
Terminé le	mercredi 7 juin 2023, 00:04
Temps mis	32 jours 9 heures
Points	9,00/9,00
Note	**10,00** sur 10,00 (**100**%)

Question 1

Correct

Note de 1,00 sur 1,00



L'alphabet de l'entrée de cet automate à pile est { a,b ✔ }.

L'alphabet de la pile est { A,B,Z ✔ }

Votre réponse est correcte.

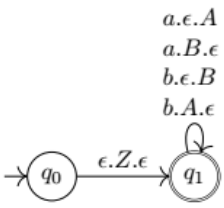
Correct

Note pour cet envoi : 1,00/1,00.

Question 2

Correct

Note de 1,00 sur 1,00



Compléter une série de transition qui permet à cet automate de reconnaitre le mot **aabb.**

format d'une configuration**: état,pile**

avec état de la forme **qi** et pile avec le symbole en haut de la pile à gauche (ne rien mettre si pile vide).

format d'une transition: **lettre consommé** (rien pour epsilon).**lettre dépilé** (rien pour epsilon)**.mot empilé** (rien pour epsilon)

Exemple: **(q3,DE)---c.D.F--->(q4,EF)**

(q0,Z ✔ **) ---**.Z. ✔ **---> (**q1, ✔ **) ---**a..A ✔ **---> (**q1,A ✔ **) ---**a..A ✔ **---> (**q1,AA ✔ **) ---**b.A. ✔ **---> (**q1,A ✔ **) ---**b.A. ✔ **---> (**q1, ✔ **)**

Votre réponse est correcte.

Correct

Note pour cet envoi : 1,00/1,00.

Question 3

Correct

Note de 1,00 sur 1,00

Considérons la grammaire qui engendre les mots sur l'alphabet $\{a, b\}$ avec autant de a que de b:

$S \rightarrow aB \mid bA \mid a \mid b \mid \epsilon$

$A \rightarrow aS \mid bAA|$

$B \rightarrow bS \mid aBB$

Prouver que cette grammaire est ambigüe en dessinant deux arbres de dérivation différents pour un même mots accepté par cette grammaire.

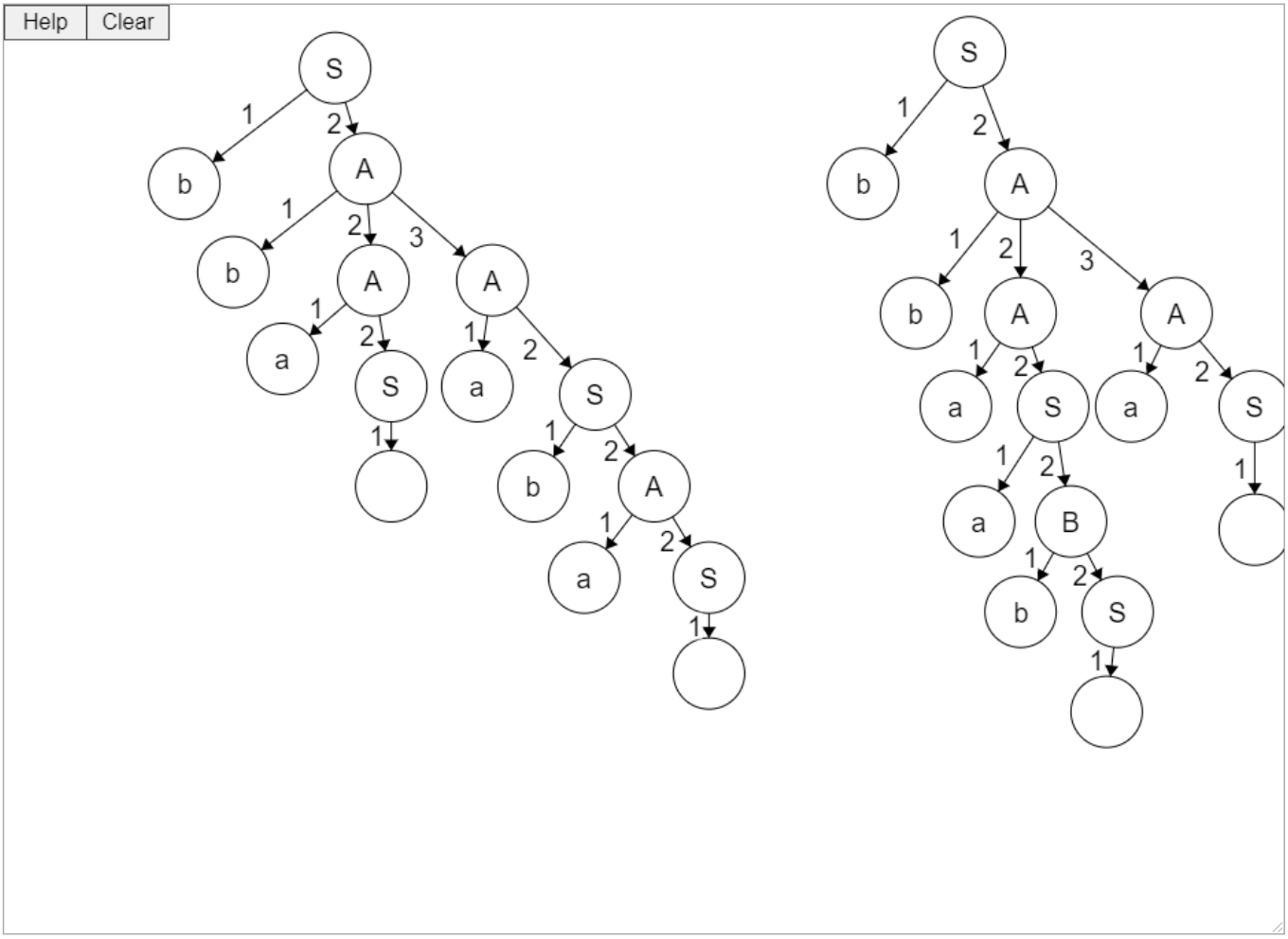
(vous devez trouver le mot vous-même)

Si un sommet a plusieurs enfant, mettre des numéros 1,2,3,4,... sur les arcs pour indiquer qui est le premier sommet lequelle est le premier, le second... Ne rien mettre dans un sommet à la place de ϵ.

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`len(arbres)`	`2`	`2`	✔
✔	`"".join(feuilles(a1)) == "".join(feuilles(a2))`	`True`	`True`	✔
✔	`respecte_regles(a1,regles,True)`	`True`	`True`	✔

	Test	Résultat attendu	Résultat obtenu	
✔	`respecte_regles(a2,regles,True)`	`True`	`True`	✔
✔	`arbre_differents(a1,a2)`	`True`	`True`	✔

Tous les tests ont été réussis ! ✔

aabbb

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.

Question 4

Correct

Note de 1,00 sur 1,00

Considérons la grammaire:

$S \rightarrow aAbS \mid bBaS \mid \epsilon$

$A \rightarrow aAbA \mid \epsilon$

$B \rightarrow bBaB \mid \epsilon$

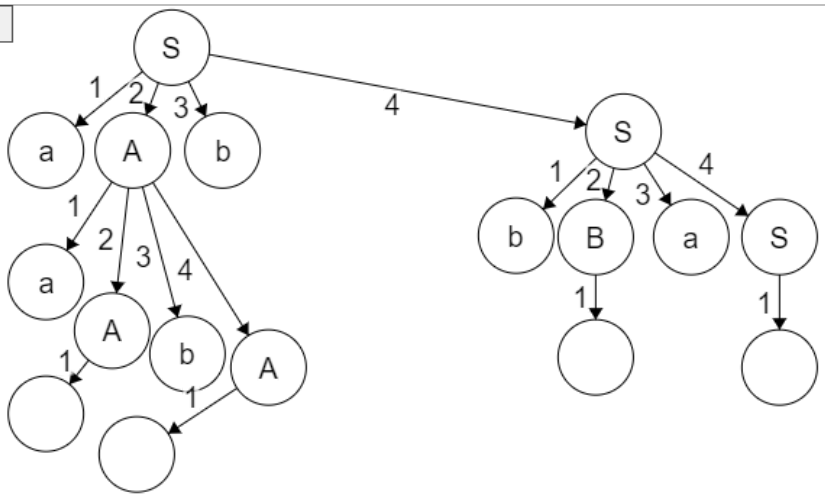
Trouver un arbre de dérivation pour le mot $aabbba$.

Si un sommet a plusieurs enfants, mettre des numéros 1,2,3,4,... sur les arcs pour indiquer qui est le premier sommet, le second, ... Ne rien mettre dans un sommet à la place de ϵ.

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`len(arbres)`	`1`	`1`	✔
✔	`"".join(feuilles(arbre))`	`aabbba`	`aabbba`	✔

	Test	Résultat attendu	Résultat obtenu	
✔	`respecte_regles(arbre,regle,True)`	`True`	`True`	✔

Tous les tests ont été réussis ! ✔

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.

Question 5

Correct

Note de 1,00 sur 1,00

Écrire un automate à pile qui reconnait les mots sur l'alphabet { **(,) , [,]** } qui sont bien parenthésés.

Le symbole sur la pile à l'état initial est **Z**.

Vous pouvez écrire vos transitions au format **a.A.w** où

- **a** est une unique lettre de l'alphabet d'entrée (ou rien pour epsilon),
- **A** est une unique lettre de l'alphabet de la pile (ou rien pour epsilon), et
- **w** est un mot sur l'alphabet de la pile.

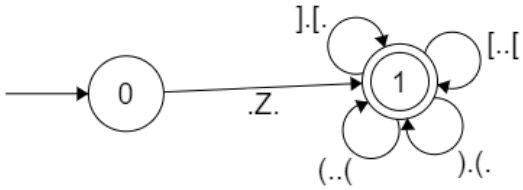
Vous pouvez écrire toutes vos transitions sur plusieurs flèches où sur une seule flèches séparés par des virgules.

Exemples: **a.A.BA,b.B.BA**

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('(())[]')`	`True`	`True`	✔
✔	`npda.accepts_input('([)]')`	`False`	`False`	✔
✔	`npda.accepts_input('((([])))')`	`True`	`True`	✔
✔	`npda.accepts_input('[(([]))]')`	`True`	`True`	✔
✔	`npda.accepts_input('([([]))]')`	`False`	`False`	✔

Tous les tests ont été réussis ! ✔

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.

Question 6

Correct

Note de 1,00 sur 1,00

On considère le langage $L = \{a^n b^m c^k \mid m = n + k\}$.

Dessiner un automate à pile qui reconnaît le langage L.

Le symbole sur la pile à l'état initial est **Z**.

Vous pouvez écrire vos transitions au format **a.A.w** où

- **a** est une unique lettre de l'alphabet d'entrée (ou rien pour epsilon),
- **A** est une unique lettre de l'alphabet de la pile (ou rien pour epsilon), et
- **w** est un mot sur l'alphabet de la pile.

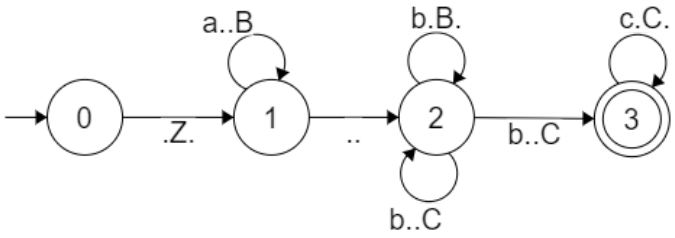
Vous pouvez écrire toutes vos transitions sur plusieurs flèches où sur une seule flèches séparés par des virgules.

Exemples: **a.A.BA,b.B.BA**

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('')`	`False`	`False`	✔
✔	`npda.accepts_input('abc')`	`False`	`False`	✔
✔	`npda.accepts_input('abbc')`	`True`	`True`	✔
✔	`npda.accepts_input('bbcc')`	`True`	`True`	✔
✔	`npda.accepts_input('ba')`	`False`	`False`	✔
✔	`npda.accepts_input('aaabbbccc')`	`False`	`False`	✔
✔	`npda.accepts_input('bbcc')`	`True`	`True`	✔
✔	`npda.accepts_input('abbbbccc')`	`True`	`True`	✔
✔	`npda.accepts_input('abbbc')`	`False`	`False`	✔

Tous les tests ont été réussis ! ✔

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.

Question 7

Correct

Note de 1,00 sur 1,00

Écrire un automate à pile qui reconnait les mots palindromes sur l'alphabet **{a,b}**.

(il doit accepter les mots de taille paire ou impaire)

Le symbole sur la pile à l'état initial est **Z**.

Vous pouvez écrire vos transitions au format **a.A.w** où

- **a** est une unique lettre de l'alphabet d'entrée (ou rien pour epsilon),
- **A** est une unique lettre de l'alphabet de la pile (ou rien pour epsilon), et
- **w** est un mot sur l'alphabet de la pile.

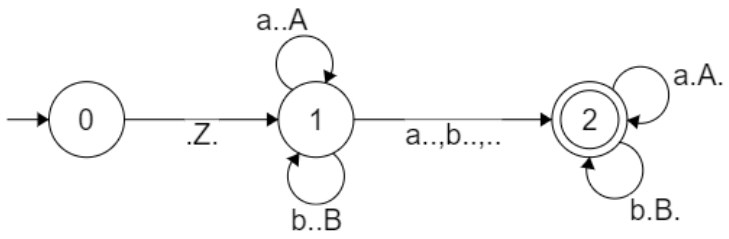
Vous pouvez écrire toutes vos transitions sur plusieurs flèches où sur une seule flèche séparées par des virgules.

Exemples: **a.A.BA,b.B.BA**

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('')`	`True`	`True`	✔
✔	`npda.accepts_input('a')`	`True`	`True`	✔
✔	`npda.accepts_input('b')`	`True`	`True`	✔
✔	`npda.accepts_input('aa')`	`True`	`True`	✔
✔	`npda.accepts_input('ab')`	`False`	`False`	✔
✔	`npda.accepts_input('ba')`	`False`	`False`	✔
✔	`npda.accepts_input('bb')`	`True`	`True`	✔
✔	`npda.accepts_input('aab')`	`False`	`False`	✔
✔	`npda.accepts_input('aba')`	`True`	`True`	✔
✔	`npda.accepts_input('baa')`	`False`	`False`	✔
✔	`npda.accepts_input('aaa')`	`True`	`True`	✔
✔	`npda.accepts_input('aaaa')`	`True`	`True`	✔

	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('aaba')`	`False`	`False`	✔
✔	`npda.accepts_input('abababa')`	`True`	`True`	✔
✔	`npda.accepts_input('abbabba')`	`True`	`True`	✔
✔	`npda.accepts_input('ababaa')`	`False`	`False`	✔
✔	`npda.accepts_input('aabbaabb')`	`False`	`False`	✔

Tous les tests ont été réussis ! ✔

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.

Question 8

Correct

Note de 1,00 sur 1,00

En utilisant l'algorithme vu en cours, transformer cette grammaire hors-contexte en automate à pile à un seul état: $Z \rightarrow aZa \mid bZb \mid a \mid b \mid \epsilon$.

un automate à pile qui reconnait les mots palindromes sur l'alphabet **{a,b}**.

Le symbole sur la pile à l'état initial est **Z**.

Vous pouvez écrire vos transitions au format **a.A.w** où

- **a** est une unique lettre de l'alphabet d'entrée (ou rien pour epsilon),
- **A** est une unique lettre de l'alphabet de la pile (ou rien pour epsilon), et
- **w** est un mot sur l'alphabet de la pile.

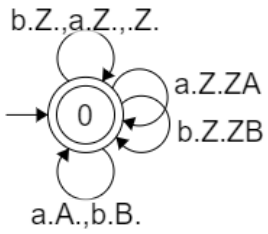
Vous pouvez écrire toutes vos transitions sur plusieurs flèches où sur une seule flèche séparées par des virgules.

Exemples: **a.A.BA,b.B.BA**

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('aabb')`	`False`	`False`	✔
✔	`npda.accepts_input('abba')`	`True`	`True`	✔
✔	`npda.accepts_input('abbaabba')`	`True`	`True`	✔
✔	`npda.accepts_input('bababab')`	`True`	`True`	✔
✔	`npda.accepts_input('baababb')`	`False`	`False`	✔
✔	`len(npda.states)`	`1`	`1`	✔

Tous les tests ont été réussis ! ✔

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.

Question 9

Correct

Note de 1,00 sur 1,00

Sur l'alphabet $\Sigma = \{1, 2, +, =\}$, on considère l'ensemble des mots représentant une égalité numérique (vraie !) sur des sommes de 1 et de 2.

Par exemple :

- $1 + 1 = 2$
- $1 + 2 = 1 + 2$
- $1 + 2 + 1 = 2 + 2$

(Pour être clair $12 + 1 = 1 + 12$ n'est pas accepté, les additions sont sur des nombres qui sont 1 ou 2).

Montrer que ce langage est hors-contexte en construisant un automate à pile qui l'engendre.

Le symbole sur la pile à l'état initial est **Z**.

Vous pouvez écrire vos transitions au format **a.A.w** où

- **a** est une unique lettre de l'alphabet d'entrée (ou rien pour epsilon),
- **A** est une unique lettre de l'alphabet de la pile (ou rien pour epsilon), et
- **w** est un mot sur l'alphabet de la pile.

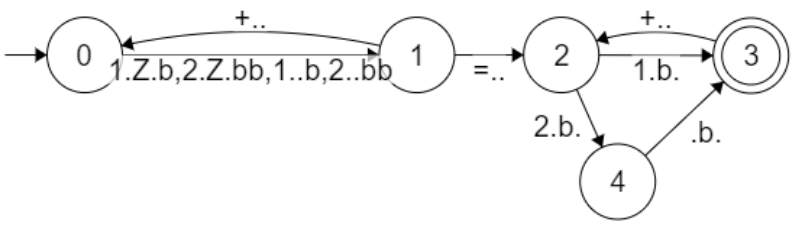
Vous pouvez écrire toutes vos transitions sur plusieurs flèches où sur une seule flèches séparés par des virgules.

Exemples: **a.A.BA,b.B.BA**

Réponse :

Réinitialiser la réponse

Help Clear



	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('=')`	`False`	`False`	✔
✔	`npda.accepts_input('+')`	`False`	`False`	✔
✔	`npda.accepts_input('1')`	`False`	`False`	✔
✔	`npda.accepts_input('1+1')`	`False`	`False`	✔
✔	`npda.accepts_input('1=1')`	`True`	`True`	✔
✔	`npda.accepts_input('2=1+1')`	`True`	`True`	✔

	Test	Résultat attendu	Résultat obtenu	
✔	`npda.accepts_input('1+1+1+1=2+2')`	`True`	`True`	✔
✔	`npda.accepts_input('2+1+1+1+1=1+2+2+1')`	`True`	`True`	✔
✔	`npda.accepts_input('2+1+1+1=2+1+1+1')`	`True`	`True`	✔

Tous les tests ont été réussis ! ✔

▸ **Montrer / masquer la solution de l'auteur de la question (Python3)**

Correct

Note pour cet envoi : 1,00/1,00.