

Exercice 1 :

On considère la syntaxe suivante :

Syntaxe	Exemple1	Exemple2	Exemple3	Exemple4
Id = valeur	Code	Code	Code	Code
Id = Id + Id	a=10	a = b + c	a=b+20	a=b*c+d+e
Id = Id + valeur				

Partie I :

On propose la grammaire suivante:

1. $S \rightarrow id = T E$
2. $T \rightarrow id$
3. $T \rightarrow valeur$
4. $E \rightarrow + T E$
5. $E \rightarrow * T E$
6. $E \rightarrow \varepsilon$

1. Donner l'ensemble V_N et l'ensemble V_T .

$V_N = \{S, T, E\}$, $V_T = \{id, =, *, +, valeur\}$

2. Donner les premiers et les suivants de chaque symbole non terminal.

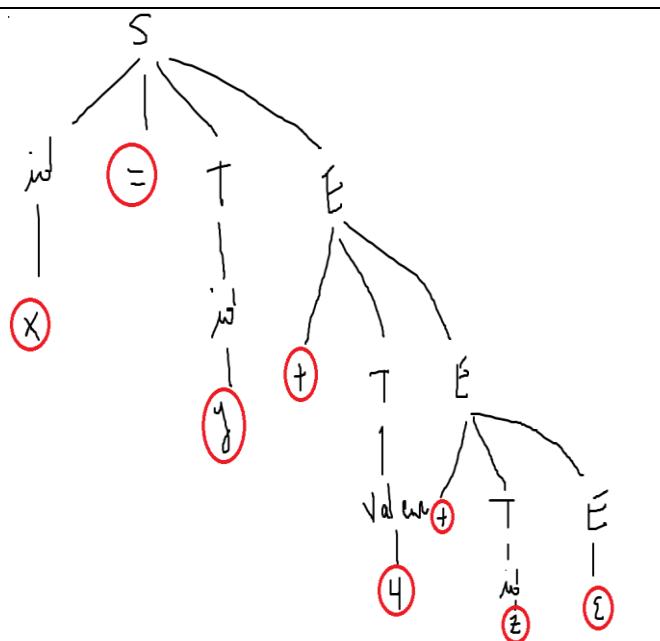
	Premier	Suivant
S	Id	\$
T	id, valeur	+, *, \$
E	+, *, ε	\$

3. Élaborer la table d'analyse LL de cette grammaire.

	id	+	*	=	valeur	\$
S	1					
T	2				3	
E		4	5			6

4. Analyser la phrase $x = y + 4 + z$ et Donnez l'arbre de dérivation associé. A noter que x,y et z sont des identificateurs.

Pile	Entrée	Sortie
\$S	$x = y + 4 + z\$$	$S \rightarrow id = T E$
\$ET=id	$x = y + 4 + z\$$	$T \rightarrow id$
\$Eid	$y + 4 + z\$$	$E \rightarrow + T E$
\$ET+=	$+ 4 + z\$$	$T \rightarrow valeur$
\$Evalueur	$4 + z\$$	$E \rightarrow + T E$
\$ET+=	$+ z\$$	$T \rightarrow id$
\$E	\$	$E \rightarrow \varepsilon$
\$	\$	Accepter



Partie II :

Cette partie traitera une grammaire permettant de produire la syntaxe d'une fonction sous Python.

On considère la syntaxe suivante:

Syntaxe	Exemple1	Exemple2	Exemple3
def fonction(liste_paramètres): instr où instr présente l'instruction de la partie I	Code def fct1(x): x=x+x*x	Code def fct2(x,y): x=x+2*y	Code def fct1(x,y,z): x=x+2*y+z
	Résultat de l'exécution	Résultat de l'exécution	Résultat de l'exécution
	>>> x=1 >>> fct1(x) >>>x 2	>>> x=1 >>> y=2 >>> fct2(x,y) >>>x 5 >>>y 2	>>> x=1 >>> y=2 >>> z=3 >>> fct3(x,y,z) >>>x 8 >>>y 2 >>>z 3

1. Proposer une grammaire pour la syntaxe des fonctions.

1. $S' \rightarrow \text{def nomFct (LP) :S}$

2. $LP \rightarrow \text{id Par}$

3. $LP \rightarrow \varepsilon$

4. $\text{Par} \rightarrow \text{,id Par}$

5. $\text{Par} \rightarrow \varepsilon$

6. $S \rightarrow \text{id} = T E$

7. $T \rightarrow \text{id}$

8. $T \rightarrow \text{valeur}$

9. $E \rightarrow + T E$

10. $E \rightarrow * T E$

11. $E \rightarrow \varepsilon$

2. Donner l'ensemble V_N et l'ensemble V_T de la nouvelle grammaire

$V_N = \{S'; LP; Par; S; T; E\}$ $V_T = \{\text{def}; \text{nomFct}; (,); ,, ::; \text{id}; =, *, +; \text{valeur}\}$

3. Donner les premiers et les suivants de chaque symbole non terminal.

	Premier	Suivant
S'	def	\$
LP	id ; ε)
Par	, ; ε)
S	id	\$
T	id ;valeur	+,*, \$
E	+ ;* ; ε	\$

4. Élaborer la table d'analyse LL de cette grammaire.

	def	nomFct	()	,	:	id	=	*	+	valeur	\$
S'	1											
LP				3			2					
Par				5	4							
S							6					
T							7				8	
E									10	9		11

Exercice 2

Un dictionnaire en Python est une structure des données qui utilisent un système d'indexation (clé) propre à lui, chaque clé est séparée de sa valeur par deux points (:), les éléments sont séparés par des virgules, et le tout est enfermé dans des accolades.

Un dictionnaire vide sans aucun article est écrit avec seulement deux accolades, comme ceci: {}.

	Script	résultat
Exemple1	>>> dict={'a':1,4:"bonjour",3.14:"pi",3:22/7}	>>> dict['a'] 1
Exemple2	>>> dict={'a':1,4:"bonjour",3.14:"pi",3:22/7}	>>> dict[4] 'bonjour'
Exemple3	>>> dict={'a':1,4:"bonjour",3.14:"pi",3:22/7}	>>> dict[3.14] 'pi'
Exemple4	>>> dict={'a':1,4:"bonjour",3.14:"pi",3:22/7}	>>> dict[3] 3.142857142857143

Partie I Analyse ascendante:

On propose la grammaire suivante:

$S \rightarrow id = \{ C \}$

$C \rightarrow id: val \mid id: val , C$

où id désigne un identificateur, et val désigne une valeur de type quelconque

- Donner l'ensemble V_N et l'ensemble V_T .

$V_N = \{ \dots \} ; V_T = \{ \dots \} ;$

- Donner les premiers et les suivants de chaque symbole non terminal.

Premier={ }	Suivant={ }
-------------	-------------

- Élaborer la table d'analyse LL de cette grammaire

.....
.....
.....
.....
.....

- Analyser l'instruction $d = \{ 'a':1 , 4: 3 \}$ et donnez l'arbre de dérivation correspondant.

Analyse			Arbre
Pile	Entrée	Sortie	

Partie II :
La deuxième partie sera dédiée à une grammaire permettant de produire un dictionnaire contenant d’autres dictionnaires

Exemple1	Exemple2
>>> d1 = {'a':1 , 4: 3.14, 7 : 9} >>> d2 = {1 : {'b':2 , 6 : "smi"} , 4: 3.14, 7 : 9} >>> d2[1] {'b': 2, 6: 'smi'} >>> d2[1]['b'] 2 >>> d2[7] 9	>>> d1 = {'a':1 , 4: 3.14, 7 : 9} >>> d2 = {1 : {'b':2 , 6 : "smi"} , 4: 3.14, 7 : 9} >>> d3 = {2 : d2 , 8: d1, 7 : 9} >>> d3[2] {1: {'b': 2, 6: 'smi'}, 4: 3.14, 7: 9} >>> d3[2][1] {'b': 2, 6: 'smi'} >>> d3[2][1][6] 'smi'

1. Proposer une grammaire pour la syntaxe permettant de produire un dictionnaire de dictionnaires
-

.....

.....

.....

.....

.....

2. Trouver l'ensemble V_N et l'ensemble V_T de la nouvelle grammaire

$V_N = \{ \}$	$V_T = \{ \}$
---------------	---------------

3. Trouver les premiers et les suivants de chaque symbole non terminal.

Premier = $\{ \}$	Suivant = $\{ \}$
-------------------	-------------------

4. Élaborer la table d'analyse LL de cette grammaire?

.....
.....
.....
.....
.....
.....