U.S.T.H.B
Faculté d'Informatique et Electronique
Licence Académique – Section A, B, C
Module de Compilation **Année Universitaire : 2019/2020**

TD N° 1 :
Analyse Lexicale

Exercice 1 :

1- Donner un automate déterministe qui reconnait un message composé d'une suite de "a" et de "b" et ne contenant pas de séquence "aaa", celle-ci indique la fin d'un message. La longueur du message est inférieure ou égale á 20 caractères.

2- Ecrire un algorithme qui reconnait un message corrte, sachant que la fin d'une chaine est un blanc et qu'une chaine est compossée d'un seul message.

Exercice 2 :

Soit la portion de programme suivante :

```
LOAD opd, cste
LOAD opd
MOVE cste, opd
STORE opd, opd
```

Où : - opd est une suite de caractères alphanumériques commençant par une lettre. La longueur ne devant pas dépasser 6 caractères.
- cste est un nombre entiers sans signe ayant au maximum 4 chiffres.

1. Donner les différentes entités lexicales de la portion de programme.
2. Construire un automate déterministe acceptant les entités de la portion de programme?
3. Développer l'analyseur lexical correspondant.

Exercice 3 :

On veut analyser le langage L défini sur $\{x, y\}^*$ et présentant les caractéristiques suivantes :
-Le langage est formé des entités de type A, B et C.
-Une entité A commence par 'x' et ne contient pas deux 'x' consécutifs.
-Une entité B commence par un 'y' qui est suivi seulement d'un nombre pair de 'x' (il y a au moins deux).
-Une entité C commence par deux 'x' ou bien deux 'y' suivis par une séquence de 'x' et de 'y' ne contenant la sous chaine 'yxy'.

1 - Construire un automate déterministe reconnaissant les entités A, B et C.
2 - Ecrire un algorithme qui fait l'analyse lexicale du langage.

Scanned by CamScanner

USTHB
Faculté d'Electronique et d'Informatique
Département d'Informatique
Licence Académique 3ème Année
Sections A,B,C
Module Compilation

Année Universitaire : 2019/2020

TD N° 2 :
Analyse Syntaxique
Méthodes Descendantes

Exercice 1 :
Factoriser les grammaires suivantes :

G1: S → S @ T |T
T →T op F | F
F → ?F | (S)|TRUE|FALSE

G2 :E→E+T|E-T|T
T→ T*F|T\F|F
F→N|(E)
N→0N|1N| ε

G3 : S→abBcA|abABc
A→cA |abB
B→Bc | caB| ε
C→Ab |S

Exercice 2 :
Eliminer la récursivité gauche dans les grammaires suivantes :

G1 : S → aAba /AB
A → Bb /a
B → Sa /c

G2 : S → Aba/Baa
A → Aab /Ab /a
B→ Bb / b

G3 : S → Ayz / Bxx / AB
A → AxB / Ay / xSz / ε
B → yB / ε

Exercice 3 :
Soit la grammaire G suivante :

S → AB | Da
A → aAb | c
B → bB | ε
D → dD | e

1- Vérifier que cette grammaire vérifie les conditions d'une grammaire LL(1).
2- Ecrire un analyseur syntaxique basé sur la descente récursive pour la grammaire.
3- Analyser la chaine ddea#.

Exercice 4 :
Soit la grammaire G suivante :

S → C $
C → E / -E
E → I / I•I
I → Id / d

a. G est-elle LL(1) ? (Justifier la réponse).
b. Eliminer la récursivité gauche dans G.
c. Factoriser la grammaire obtenue en b.
d. Calculer les ensembles Debut et Suivant de la grammaire obtenue en c.
e. Construire la table d'analyse LL(1) de la grammaire obtenue en c.
f. La grammaire obtenue en c est-elle LL(1) ? Justifier.
g. Analyser la chaîne : -d•d$#.

Scanned by CamScanner

USTHB
Faculté d'Electronique et d'Informatique
Département d'Informatique
Licence Académique
Sections A,B,C
Module Compilation

Année Universitaire : 2019/2020

TD N° 3 :
Analyse Syntaxique
Méthodes Ascendantes

Exercice 1 :

Les grammaires suivantes sont-elles LR(1)?SLR(1) ?

G1 : $M \rightarrow NM / y$
$N \rightarrow MN / x$

G2 : $S \rightarrow A1A0 / B0B1$
$A \rightarrow \varepsilon$
$B \rightarrow \varepsilon$

Exercice 2 :

On considère la grammaire G suivante :

$S \rightarrow Aa \mid bAc \mid dc \mid bda$
$A \rightarrow d$

1- G est-elle SLR(1)? Justifiez.
2- G est-elle LR(1)? Justifiez.
3- G est-elle LALR(1)? Justifiez
4- Comparer les trois tables d'analyse ?

Exercice 3 :

Soit la grammaire G suivante :

$E \rightarrow !E / E\$E / E\&E / nb$

1 G est-elle ambiguë ? est-elle LR(1) ?
2 Construire la table d'analyse SLR(1) .
Peut-on éliminer les multidéfinitions en adoptant certaines conventions ? Si oui, analyser la chaine a&b$a#.

Scanned by CamScanner

USTHB
Faculté d'Electronique et d'Informatique
Département d'Informatique
Licence Académique
Sections A,B,C
Module Compilation

Année Universitaire : 2019/2020

TD N° 4 :
Formes Intermédiaires

Exercice 1 :

Traduire les expressions suivantes en code postfixé, quadruplés, et arbres abstraits :

1- a:=(a+b*c)+((d+b*c)-f*g)+f*g*d
2- a:=a+b*((d+a)/(d+b))*c

Exercice 2:

Traduire les expressions suivantes en code postfixé, quadruplés et arbres abstraits :

```
1.Begin  Integer Array B[x:x+y];
         Repeat J:= J+1;
                x := B[x+1]*4;
         Until (J<=20) or (x>10);
 End;

2.Begin If (a<b*c)
            Then While a<=b*c
                    Do Begin a:=a+1;
                             a:=b*c;
                       End;
            Else  a:=b*c;
 End;
```

Exercice 3:

Traduire l'expression booléenne suivante en quadruplés en utilisant les opérateurs BZ, BNZ, BR et :=

V :=X and Y or Z and Y or not X

Scanned by CamScanner

USTHB
Faculté d'Electronique et d'Informatique
Département d'Informatique
Licence Académique
Sections A,B,C
Module Compilation

Année Universitaire : 2019/2020

TD N° 5:
Traduction dirigée par la syntaxe

Exercice 1:
Soit l'instruction suivante :
Execute <Inst> Provided <cond>

Sémantique:
L'instruction est exécutée si la condition est vérifiée.

Donner le schéma de traduction sous forme postfixée en utilisant un analyseur descendant.

Exercice 2:
Soit l'instruction suivante :
Si < exparithm> = neg <suite1> nul <suite2> pos <suite3> finisi

Sémantique:
Si exparithm est négative, exécuter suite1 et aller à finisi, sinon si exparithm est nul, exécuter suit2 et aller à finisi, si elle est positive, exécuter suite3.

1. Donner la grammaire syntaxique.
2. Donner le schéma de traduction sous forme de quadruplés avec une analyse descendante.

Exercice 3:
Soit l'instruction :
Id := moyenne (<exp1>,<exp2>, … <expn>)
1. Donner la grammaire syntaxique.
2. Donner le schéma de traduction sous forme de quadruplés avec une analyse ascendante.

Exercice 4:
Soit l'instruction suivante :
Id := SUM (<exp1>,<exp2>,...,<expn>)

Sémantique:
L'identificateur Id reçoit la somme des expressions strictement positives parmi les n expressions données entre parenthèses.Si aucun expression n'est positive, Id reçoit 0.

1. Donner la grammaire syntaxique.
2. Donner le schéma de traduction sous forme de quadruplés, dans le cas d'une analyse asscendante.

Scanned by CamScanner