ANNEE UNIVERSITAIRE 2007/2008
1ERE SESSION DE PRINTEMPS

Parcours : CSB6
Code UE : INF354
Epreuve : Analyse Syntaxique
Date : Mardi 22 Avril 2008
Heure : 8H30
Durée : 1H30
Documents : autorisés
Epreuve de : Mme Frédérique Carrère

Les exercices sont indépendants.

Exercice 1

Pour cet exercice, on définit un *mot* comme une suite de une ou plusieurs lettres minuscules ou majuscules, à l'exclusion de tout autre caractère.
Ecrire un analyseur lexical, dans le format reconnu par `lex`, qui affiche sur la sortie le texte fourni en entrée, en remplaçant l'abréviation *ssi* par *si et seulement si*. Attention, l'analyseur ne doit pas remplacer le motif *ssi* s'il figure à l'intérieur d'un mot (exemple : *mission*, *aussi*). La fonction `yylex()` devra afficher à la fin du traitement le nombre d'occurrences remplacées.

Exercice 2

Soit la grammaire G suivante, écrite dans le format reconnu par `yacc` :

```
%token ID
%%

exp  :  exp  '<' exp
     |  exp  '=' exp
     | ID
     ;

%%
```

Le token `ID` représente un identificateur.

1. Construisez l'automate des $LR(0)$-items, en indiquant pour chaque état la liste complète de ses items.
2. Quelle sorte de conflits apparaissent ?
3. Proposez une solution pour résoudre ces conflits. Indiquer les modifications à apporter dans le fichier `yacc`.

Exercice 3

Soit G la grammaire des listes suivante écrite dans le format reconnu par `yacc`, :

```
%token ID
%%
S  :  '(' L ',' S ')'
   | A
   ;
```

```
L  :  L ’,’  S
   |  S
   ;

A  :  ’(’ A ’)’
   | ID
   ;
%%
```

L'automate des $LR(0)$-items est donné en annexe.

1. La grammaire G est-elle $LR(0)$? Pourquoi ?
2. Calculer les ensembles $Suivant(X)$ pour tout non-terminal X de la grammaire.
3. La grammaire G est-elle $SLR(1)$? Pourquoi ?

Exercice 4

Soit G la grammaire suivante écrite dans le format reconnu par `yacc`, :

```
%token ID CHAR INT DOUBLE STRUCT NUM
%%
declar : type ID tab ’;’ { printf("Declaration de la variable %c \n", $2); }
       ;

type : CHAR
     | INT
     | DOUBLE
     | STRUCT ’{’ list ’}’
     ;

list : list declar
     |
     ;

tab : ’[’ NUM ’]’
    |
    ;
%%
```

Afin de réserver l'espace mémoire correct dans la pile lorsqu'un identificateur est déclaré, on veut connaître la taille en octets de chaque variable.
On suppose qu'un identificateur est formé d'une seule lettre de l'alphabet. Ajoutez pour chaque règle de grammaire, un calcul d'attributs qui permette de connaître la taille en octets des variables déclarées. Modifiez l'instruction `printf` pour que l'analyseur affiche après le nom de la variable, le nombre d'octets nécessaires à son initialisation.
On considèrera qu'un entier ou un caractère occupe 1 octet, un réel double occupe 4 octets. Une variable de type structure occupe la place nécessaire pour tous les champs de la structure. Une variable de type tableau de taille n occupe la place nécessaire pour les n cases du tableau.
On supposera que les attributs des token `NUM` et `ID` ont été convenablement initialisés dans le fichier `lex`, l'attribut du token `NUM` étant la valeur de l'entier lu par `yylex()`.

Annexe :

```
   0  $accept : S $end
   1  S : '(' L ',' S ')'
   2    | A
   3  L : L ',' S
   4    | S
   5  A : '(' A ')'
   6    | ID

state 0
$accept : . S $end  (0)

ID  shift 1
'('  shift 2
.  error

S  goto 3
A  goto 4

state 1
A : ID .  (6)

.  reduce 6

state 2
S : '(' . L ',' S ')'  (1)
A : '(' . A ')'  (5)

ID  shift 1
'('  shift 2
.  error

S  goto 5
L  goto 6
A  goto 7

state 3
$accept : S . $end  (0)

$end  accept

state 4
S : A .  (2)

.  reduce 2
```

```
state 5
L : S .  (4)

.  reduce 4

state 6
S : '(' L . ',' S ')'  (1)
L : L . ',' S  (3)

','  shift 8
.  error

state 7
S : A .  (2)
A : '(' A . ')'  (5)

')'  shift 9
','  reduce 2

state 8
S : '(' L ',' . S ')'  (1)
L : L ',' . S  (3)

ID  shift 1
'('  shift 2
.  error

S  goto 10
A  goto 4

state 9
A : '(' A ')' .  (5)

.  reduce 5

state 10
S : '(' L ',' S . ')'  (1)
L : L ',' S .  (3)

')'  shift 11
','  reduce 3

state 11
S : '(' L ',' S ')' .  (1)

.  reduce 1
```

```
6 terminals, 4 nonterminals
7 grammar rules, 12 states
```